UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

GESTIÓN DE AULAS DE EXAMEN

Verónica Morata Perea

Tutor: Fernando Díez

JULIO 2014

Resumen

La implantación desde 2009 de las nuevas titulaciones de Grado, de acuerdo con el nuevo marco educativo establecido a través del Espacio Europeo de Educación Superior (EEES), ha supuesto un crecimiento notable en la oferta de títulos en las distintas Universidades del estado español. En particular, en la Universidad Autónoma de Madrid, la oferta académica de títulos es cercana al centenar. En estas circunstancias, el aprovechamiento de los recursos de la Universidad es una máxima para cualquier equipo de gestión bien al nivel de centros (Facultades y Escuela) e, inclusive, al nivel de la propia Universidad. Por tanto, el objetivo que se ha pretendido realizar mediante este Trabajo Fin de Grado ha sido el de crear una aplicación que facilite y de soporte a la gestión de las sesiones de examen, con el propósito de automatizar toda la funcionalidad que supone la realización de múltiples pruebas de evaluación coincidentes en el espacio y en el tiempo. La implantación de sistemas de gestión y control como el que proponemos posibilitaría la coincidencia de pruebas de distintas asignaturas en un mismo aula, permitiendo un aprovechamiento de los recursos, materiales y humanos, más adecuado que en la actualidad.

En el procedimiento actual no se llevan a cabo tareas como el control de estudiantes en la entrada y salida de la sesión, la realización de exámenes de distintas asignaturas en el mismo aula o la asignación de los asientos en el aula, entre otras. Todas estas funciones se podrían realizar con la aplicación que se ha desarrollado en este proyecto: “Gestión de aulas de examen”. Para ello se ha realizado un estudio de distintas tecnologías para averiguar cuáles se adaptaban mejor a este proyecto, además de analizar los aspectos más importantes a tener en cuenta a la hora de realizar una sesión de exámenes. Por otra parte se precisa que esta gestión sea fluida, lo que requiere aportar una interfaz intuitiva a la aplicación y el uso de un lector de código de barras que agilice la entrada y salida de los estudiantes de la sesión.

La aplicación desarrollada está orientada al uso por parte de los miembros de un tribunal, desde el momento que se inicia la sesión de examen hasta que concluye.

En su estado actual constituye una aplicación piloto de posibles proyectos de innovación docente futuros.

Palabras clave: Gestión, Examen, Aula, Sesión, Estudiante, Aplicación, Agilidad, .NET, C#, Framework, Modelo-Vista-Controlador.

Abstract

The implementation since 2009 of the new degree, according to the new educational framework established by the European Higher Education Area (EHEA) has been a remarkable growth in the supply of securities in the various universities of the Spanish state. In particular, the Autonomous University of Madrid, the academic offerings of securities is close to hundred. In these circumstances, the use of University resources is a maximum for any management team either at the level of schools (Colleges and School), and even at the level of the university. Therefore, the objective has sought performed through this Bachelor Thesis was to create an application that facilitates and supports management review sessions, in order to automate all the functionality that involves making multiple matching evaluation tests in space and in time. The implementation of management and control systems such as we propose would allow matching tests in different subjects in the same classroom, allowing use of resources, material and human, more appropriate today.

In the current procedure does not perform tasks like checking students in and out of the meeting, the holding of examinations in different subjects in the same classroom or the allocation of seats in the classroom, among others. All these functions could be performed with the application that has been developed in this project: "Managing examination rooms." This has made a study of various technologies to determine which are best suited to this project, in addition to analyzing the most important aspects to consider when performing a test session. Furthermore it is stated that this management is fluid, which requires providing an intuitive interface to the application and use of a barcode reader to expedite the entry and exit of students in the session.

The developed application is geared for use by members of the court, from the time the review session begins until concludes.

In its current state is a pilot application of possible future teaching innovation projects.

Index Terms: Management, Exam, Classroom, Session, Student, Application, Agility .NET, C#, Framework, Model-View-Controller.

Agradecimientos

En primer lugar agradecer a mi tutor Fernando Diez, por confiar en mí para realizar este proyecto, y por toda la ayuda que me ha ofrecido durante su desarrollo desde el principio hasta el final, muchas gracias.

A todos aquellos que me he encontrado en mi camino durante estos años, y habéis hecho que esto se lleve con más alegría y por los buenos momentos que hemos pasado juntos, vosotros lo sabéis.

A las tres de siempre, porque sin vosotras no hubiera sido lo mismo dentro y fuera de la universidad y por supuesto a la rubia, que durante este último año hemos hecho muy buenas migas y me ha ayudado desde el minuto cero.

Y lo más importante a mis padres, mi hermana y toda mi familia por el apoyo incondicional en los buenos y malos momentos durante todos estos años, por no dejar que me rindiera en los momentos de flojera, porque sin vosotros hubiese sido imposible. Muchas gracias.

Índice de contenidos

Índice de tablas

Índice de Figuras

Glosario

TIC Tecnologías de la información y la comunicación

ASP.NET Framework para aplicaciones web desarrollado y comercializado por Microsoft.

CSS Siglas de Cascading Style Sheets - Hojas de Estilo en Cascada - que es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación

PDF formato de almacenamiento de documentos digitales independiente de plataformas de software o hardware

SQL Lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas

UNED Universidad Nacional de Educación a Distancia, es una universidad pública española de ámbito estatal

Framework (Marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Bootstrap Framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

Webservice Tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Aplicación Tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajos.

1 Introducción

En los últimos años los avances tecnológicos han ido creciendo a pasos vertiginosos, dando lugar a numerosas aplicaciones que son usadas a diario, ya que nos proporcionan facilidad a la hora de resolver problemas y desempeñar actividades.

Desde antiguamente, se ha llevado a cabo la realización de exámenes para evaluar los distintos conocimientos adquiridos por una persona sobre cualquier materia. El término examen está vinculado al concepto de evaluación que hace referencia a calcular el valor de lo aprendido. En la actualidad, existen diversos tipos de examen, como pueden ser: oral, escrito o físico. La popularización de la enseñanza, y en muchos casos la masificación a la hora de la realización de evaluaciones, conlleva problemas logísticos de importancia. El aprovechamiento óptimo del espacio físico de que disponen las organizaciones que realizan las pruebas, exige una planificación cuidadosa.

Es por eso, que hemos desarrollado una aplicación basada en las TIC que facilite el trabajo a la hora de organizar las sesiones de exámenes, bien en la universidad o en cualquier otro ámbito en el que sea necesaria la evaluación.

Para la realización de esta aplicación, nos centraremos en los exámenes escritos, que conllevan su ejecución en un determinado lugar, donde se concentran todas las personas que van a ser examinadas, por lo que debe existir un control sobre éstas, para que la sesión del examen se desarrolle con normalidad.

Sobre esta idea se basa el proyecto “Gestión de aulas de examen”. En la actualidad no se utiliza ninguna herramienta que controle digitalmente todo el proceso de gestión de los exámenes, lo que provoca que sea un proceso complejo que requiere una planificación exhaustiva. Existen diversos parámetros que condicionan dicho proceso como el número de estudiantes matriculados, la capacidad de las aulas, etc. Para poder llevar un control regulado de las sesiones de exámenes hemos implementado esta aplicación mediante el

desarrollo de los medios TIC, que permitan llevar un seguimiento automatizado de los estudiantes en la entrada, el desarrollo y la salida del examen. El avance de las TIC posibilita hacer una gestión más eficiente del control de exámenes. Mediante un control más eficiente se garantizarían aspectos relacionados directamente con la evaluación como, la identificación y autentificación del estudiante mediante su carnet universitario, el uso de material permitido, la colocación del alumno dentro del aula, etc.

La aplicación está destinada para el uso de las personas que pertenecen al tribunal, el cual está formado por miembros del profesorado de la universidad, que se encargarán de controlar el desarrollo de todas las sesiones de examen de una determinada convocatoria.

1.1 Estructura del documento

En esta memoria se recogen las fases principales del desarrollo de un proyecto: análisis, diseño, desarrollo e implementación, pruebas y mantenimiento

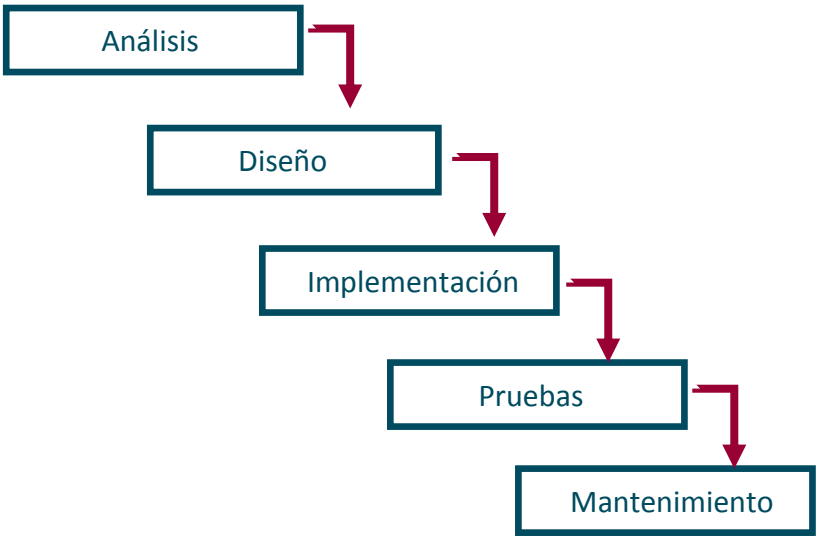


Figura: Ciclo de vida

En el siguiente capítulo, se ha realizado una breve revisión del Estado del arte, para comprobar qué aplicaciones podemos encontrar actualmente que tengan un funcionamiento similar.

En el capítulo **3** describiremos los objetivos que se intentan alcanzar con el desarrollo de este proyecto.

Posteriormente, en el capítulo **4,** trataremos las tecnologías empleadas para el desarrollo de la aplicación "Gestión de aulas de examen".

En el capítulo **5**, se recoge el análisis de requisitos fundamentales para la aplicación.

En el capítulo **6,** abordamos el diseño del proyecto, especificando cuáll es su funcionalidad en cada una de las fases de la aplicación.

A continuación en el capítulo **7**, describimos la implementación del diseño realizado en el lenguaje de programación elegido.

En el capítulo 8, se detallarán las pruebas realizadas a la aplicación así como cuáles han sido sus resultados.

En el capítulo **9**, se analizará cuál debería ser el mantenimiento de la aplicación, para que ésta sesté acorde con las circunstancias que van sucediendo.

Por último en los capítulos **10** y **11** se presentarán las conclusiones una vez realizado el proyecto, y las líneas futuras como posibles mejoras que se puedan aportar al proyecto.

2 Estado del arte

En la actualidad, en la Escuela Politécnica de la Universidad Autónoma de Madrid, no se dispone de ninguna aplicación, que como hemos dicho anteriormente, permita llevar una gestión más eficiente del control de exámenes.

El proyecto que hemos desarrollado está basado en la herramienta "**Valija Virtual**", desarrollada y utilizada en la **UNED.** El modelo "Valija" de la UNED, es una herramienta pionera, desarrollada en el Centro Asociado de Barbastro con el fin de simplificar los procesos de preparación, clasificación, transporte, reparto y recogida de exámenes, proporcionando una mayor seguridad a todo el proceso.

En el presente, este sistema informático es utilizado en todos los centros de la UNED para la realización de las pruebas presenciales. Con la aplicación "Valija virtual", todo el proceso de gestión de pruebas presenciales es controlado digitalmente, aportando grandes ventajas, como lo son: minimizar costes del papel, mejora la gestión del tiempo, facilita las labores administrativas y la labor del Tribunal en las pruebas presenciales, entre las más destacables.

Los resultados del uso de ésta herramienta son muy positivos como podemos ver en [1], lo cual representa un incentivo a la hora de desarrollar una aplicación de características semejantes que pueda ser utilizada en la universidad.

Para ver todos los detalles de la aplicación "Valija Virtual" podemos acceder al siguiente enlace [2].

3 Objetivos

Un proyecto de las características del que planteamos conlleva numerosos aspectos relacionados con la gestión de la realización de pruebas presenciales. La virtualización de las pruebas requiere controlar aspectos relativos al tribunal, a los estudiantes, las convocatorias, las sesiones, las aulas con sus restricciones de espacio, la coincidencia de exámenes, el material de que pueden hacer uso los estudiantes, etc. Como se observa son numerosos y de naturaleza muy dispar los elementos que deben controlarse.

En líneas anteriores ya hemos indicado que el objetivo principal del desarrollo de esta aplicación es: facilitar la gestión de los exámenes presenciales, permitiendo el control de los mismos por miembros del tribunal de una manera sencilla y eficaz.

Este objetivo principal se puede descomponer en una serie de objetivos parciales que enumeramos a continuación:

- **Aprovechamiento de espacios.** Uno de los objetivos más importantes, consiste en conseguir que se puedan examinar en un mismo aula, alumnos que se presentan a distintas asignaturas, inclusive de diferentes cursos y carreras.
- **Autenticación de estudiantes.** Realizar un registro de la entrada y salida de los estudiantes en la sesión de examen, que facilite la lectura de los asistentes y comprobar ciertos datos de la sesión una vez haya finalizado.
- **Gestión del aula.** Se facilita la labor de asignar los asientos a cada estudiante dentro del aula, puesto que la aplicación los posicionará automáticamente, dependiendo de la estrategia elegida por el tribunal para su distribución. Todo ello con una interfaz clara e intuitiva, que agilice aun más todo el proceso.
- **Administración de incidencias**. Permitir la anotación de problemas que puedan surgir a lo largo del desarrollo de la sesión, por ejemplo, casos de copia, material, llegada tarde a la entrada, examen no legible, falta pegatina,
- **Estadísticas.** Ofrecer la visión de información acerca de la sesión, como el número de asignaturas y la cantidad de alumnos matriculados, o conocer la cifra de alumnos de cada asignatura que se encuentran en el aula.

4 Tecnologías empleadas

Para tomar la decisión sobre qué tecnologías se usarían para desarrollar el proyecto, previamente hemos realizado un análisis entre los distintos lenguajes y modelos de desarrollos de aplicaciones web.

Hemos basado el análisis entre **PHP** y **ASP.NET**, que analizamos a continuación.

PHP, proviene de *Pre Procesador de Hipertexto*. Es un lenguaje interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente para la interpretación del lado del servidor, pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica. Podemos comprobar más detalles sobre PHP en [3]

ASP.NET, es un modelo de desarrollo Web unificado que incluye los servicios necesarios para crear aplicaciones Web empresariales con el código mínimo. ASP.NET forma parte de .NET Framework y al codificar las aplicaciones ASP.NET tiene acceso a las clases en .NET Framework. El código de las aplicaciones puede escribirse en cualquier lenguaje compatible con el "Common Language Runtime" (CLR), entre ellos Microsoft Visual Basic y C#. Podemos comprobar más detalles sobre ASP.NET en [4].

Mientras que PHP es un lenguaje orientado al desarrollo web, .NET sin embargo permite construir desde aplicaciones de escritorio, aplicaciones móviles o incluso hacer un *webservice.* Si la aplicación está bien diseñada, en la mayoría de las ocasiones basta con cambiar la capa de presentación para hacerla compatible.

Las ventajas que ofrece .NET, es que es idóneo si lo que queremos es sacarle partido a un gran sistema operativo como Windows que tiene multitud de aplicaciones. Está preparado para gestionar usuarios del directorio activo de Windows (permite crear grupos de usuarios), e integrarse con Microsoft Office (lo que permitiría insertar las notas en una hoja excel), entre otras ventajas.

Para el desarrollo del proyecto, después de valorar las ventajas y los contras de ambos, hemos optado por utilizar **ASP.NET** con el framework **MVC** [5]**,** es decir usando el modelo vista-controlador y el lenguaje **C#**, ya que hemos visto que era el modelo de desarrollo que mejor se adaptaba a las necesidades que precisábamos. Es por eso que a continuación profundizamos aun más en este modelo.

ASP.NET introduce el concepto del "*code-behind",* por el que una misma página se compone de dos ficheros: el de la interfaz de usuario y el código del controlador. Con ello se facilita la programación de aplicaciones en múltiples capas, lo que en definitiva se traduce en la total separación entre lo que el usuario ve y lo que la base de datos tiene almacenado, por lo que cualquier cambio drástico de especificaciones minimiza los cambios en la aplicación y maximiza la facilidad de mantenimiento.

Pasaremos a hablar del modelo vista-controlador más adelante en el apartado de implementación.

Para el desarrollo del proyecto usaremos "**Visual Studio 2013**" [6], ya que nos facilita la organización de la aplicación.

A continuación enumeramos algunas de las carpetas en las que se divide el proyecto con la utilización de "Visual Studio 2013":

- **App_Start**, contiene los archivos de código que se ejecutan al inicializar la aplicación. Esta clase es la encargada de iniciar la aplicación, el directorio App_Start está pensando para ubicar las clases de configuración para el inicio de la aplicación.
- **Content**, contiene los ficheros ".**css",** los cuales se encargan de dar cuerpo a la interfaz de la aplicación.
- **Controlers,** es el lugar para los controladores, que son las clases encargadas de recibir y gestionar las peticiones http de la aplicación.
- **Models,** es la ubicación que nos propone ASP.NET MVC para las clases que representan el modelo de la aplicación, los datos que gestiona nuestra aplicación.
- **Scripts,** está pensado para ubicar los archivos de javascript (*.js). El código javascript es ejecutado en el contexto del navegador, es decir, en la parte cliente, y nos permite ejecutar acciones sin necesidad de enviar los datos al servidor.

- **Views,** contiene los archivos de vista. Al patrón MVC los controladores devuelven vistas sobre las que inyectamos el modelo de nuestra aplicación. Estas vistas son interpretadas por el motor de renderización, "Razor" en nuestro caso. Son archivos similares a aplicaciones de ASP clásico, donde tenemos código HTML estático y determinadas zonas de código que son ejecutadas en el servidor.

Además se incluyen los siguientes archivos de "*layout*" en nuestro proyecto:

- **Web.config,** es el archivo principal de configuración de ASP.NET. Se trata de un archivo XML donde se define la configuración de la aplicación.
- **Global.asax**, es una instancia de una clase derivada de System.Web.HttpApplication. Esta clase es el punto de entrada de nuestra aplicación.

5 Análisis de requisitos

En este capítulo se realiza un análisis de los requisitos que debe tener la aplicación para su correcto funcionamiento, para ello planteamos la funcionalidad y características del proyecto divididas en requisitos funcionales y no funcionales.

5.1 Requisitos funcionales

Los requisitos funcionales son aquellos que describen la funcionalidad que tiene la aplicación, es decir responden a la pregunta "¿Qué hace?". Para su definición hemos dividido el proyecto en distintos subsistemas:

5.1.1 Sesiones

En este subsistema, se exponen los requisitos que están relacionados con la gestión de las sesiones de examen de la aplicación.

Identificador	RF1.1
Nombre	Crear nueva sesión
Prioridad	Alta
Descripción	Será necesaria la creación de sesiones, para poder después gestionar cada una de ellas con el uso de la aplicación.
Pasos	▪ Debemos de insertar la ruta adecuada, (/Sesiones) para acceder a la lista de sesiones. ▪ Seleccionar el enlace "Create New", tras el cual aparecerá un formulario. ▪ Rellenar el formulario con los campos de fecha y hora de la sesión. ▪ Pulsar el botón "Create", que nos redirige al índice de sesiones, donde aparecerá la nueva sesión.

Tabla Requisitos Funcionales 1: RF1.1

Identificador	RF1.2
Nombre	Modificar sesión
Prioridad	Alta
Descripción	Es necesario permitir la modificación de una sesión, en caso de error a la hora de crearla, o modificación por algún motivo de peso.
Pasos	▪ Debemos que insertar la ruta adecuada, (/Sesiones) para acceder a la lista de sesiones. ▪ Seleccionar el enlace “Edit”, tras el cual aparecerá un formulario. ▪ Modificar los datos que se desee. ▪ Pulsar el botón “Save”, que nos redirige al índice de sesiones, donde aparecerá la sesión modificada.

Tabla Requisitos Funcionales 2: RF1.2

Identificador	RF1.3
Nombre	Eliminar sesión
Prioridad	Alta
Descripción	Será necesaria la eliminación de sesiones, en caso de crear una sesión ya existente, o por cualquier otro motivo de peso.
Pasos	▪ Debemos que insertar la ruta adecuada, (/Sesiones) para acceder a la lista de sesiones. ▪ Seleccionar el enlace “Delete”, tras el cual aparecerá la información de la sesión seleccionada. ▪ Pulsar el botón “Delete”, que nos redirige al índice de sesiones.

Tabla Requisitos Funcionales 3: RF1.3

Identificador	RF1.4
Nombre	Consultar sesiones del sistema
Prioridad	Media
Descripción	Será útil poder consultar las sesiones que ya están creadas en el sistema
Pasos	▪ Debemos que insertar la ruta adecuada, (/Sesiones) para acceder a la lista de sesiones, en la cual nos aparecerá una lista con las sesiones existentes en la base de datos.

Tabla Requisitos Funcionales 4: RF1.4

Identificador	RF1.5
Nombre	Abrir sesión de examen
Prioridad	Alta
Descripción	Es imprescindible poder abrir una sesión, para comenzar su gestión con la aplicación.
Pasos	▪ Un miembro del tribunal debe iniciar sesión con su usuario y contraseña. ▪ Después aparecerá un desplegable donde elegirá la sesión que quiere abrir, indicando la fecha y hora de la sesión. ▪ A continuación se debe pulsar el botón “Abrir Sesión”, quedando la sesión seleccionada abierta para su gestión.

Tabla Requisitos Funcionales 5: RF1.5

Identificador	RF1.6
Nombre	Cerrar sesión de examen
Prioridad	Alta
Descripción	Será necesaria la creación de sesiones, para poder después gestionar cada una de ellas con el uso de la aplicación.
Pasos	▪ Previamente serán necesarios realizar los pasos indicados en la tabla anterior RF1.5. ▪ A continuación, pasaremos a la elección del aula y la estrategia para la colocación en la misma. ▪ Después, comenzaran a entrar los estudiantes al aula, una vez se termine la entrada de estudiantes, se procede a la entrega de exámenes. ▪ Cuando ya están todos los exámenes entregados, pulsando el botón “Cerrar sesión”, la sesión quedará finalizada.

Tabla Requisitos Funcionales 6: RF1.6

Identificador	RF1.7
Nombre	Consultar datos de la sesión
Prioridad	Media
Descripción	Puede resultar útil conocer los datos de la sesión que se está realizando, por ejemplo el número total de alumnos de la sesión, o el número de alumnos de cada asignatura de la sesión
Pasos	▪ Un miembro del tribunal, debe iniciar una sesión, como hemos indicado anteriormente. ▪ Una vez lleguemos a la pantalla de entrada de estudiantes, se podrá pulsar el botón “Estadisticas”, para consultar datos sobre la sesión. ▪ Este botón también está disponible en la pantalla de entrega de exámenes.

Tabla Requisitos Funcionales 7: RF1.7

5.1.2 Aulas

En este subsistema se presentan las funciones que están vinculadas a las aulas donde se desarrollan las sesiones de examen.

Identificador	RF2.1
Nombre	Crear nueva aula
Prioridad	Alta
Descripción	Será necesaria la creación de aulas, para la realización de las sesiones de examen. Además pueden habilitarse nuevas aulas que en la actualidad no existen.
Pasos	▪ Sera necesario introducir la ruta correcta (/Aulas), para obtener la lista de aulas. ▪ Una vez obtengamos la lista de aulas, debemos presionar sobre el enlace “Create New”, que nos llevará a un formulario para crear una nueva aula. ▪ Cuando tengamos completos todos los campos pulsamos el botón “Create”, y ya tendremos la nueva aula.

Tabla Requisitos Funcionales 8: RF2.1

Identificador	RF2.2
Nombre	Modificar datos del aula
Prioridad	Alta
Descripción	Debe ser posible la modificación de las aulas, ya que se pueden realizar reformas en los edificios y así modificar las aulas de los mismos.
Pasos	▪ Seguiremos los mismos pasos para obtener la lista de aulas, tal y como indicamos en la tabla anterior (*Tabla de Requisitos Funcionales 8).* ▪ Una vez tenemos la lista pulsaremos el enlace "Edit", el cual nos mostrará un formulario con los datos actuales del aula, permitiendo su modificación. ▪ Cuando hayamos modificado aquello que necesitábamos pulsaremos el botón "Save", guardando los cambios realizados.

Tabla Requisitos Funcionales 9: RF2.2

Identificador	RF2.3
Nombre	Eliminar aula
Prioridad	Media
Descripción	De la misma forma que en la tabla anterior, pueden eliminarse aulas que han desaparecido por alguna reforma, o por algún motivo se ha decidido que ya no son útiles para realizar las sesiones.
Pasos	▪ Seguiremos los mismos pasos para obtener la lista de aulas, tal y como indicamos en la tabla anterior (*Tabla de Requisitos Funcionales 8).* ▪ Una vez tenemos la lista pulsaremos el enlace "Delete", el cual nos mostrará los datos del aula que vamos a eliminar. ▪ En la misma página pulsaremos el botón "Delete", y el aula se borrará de la base de datos.

Tabla Requisitos Funcionales 10: RF2.3

Identificador	RF2.4
Nombre	Consultar aulas del sistema
Prioridad	Media
Descripción	Puede resultar necesario consultar el número y el tamaño de las aulas para conocer el espacio del que se dispone para la realización de las sesiones
Pasos	▪ Para consultar las aulas almacenadas en la base de datos, basta con introducir la ruta correcta, (/Aulas), como hemos indicado anteriormente.

Tabla Requisitos Funcionales 11: RF2.4

Identificador	RF2.5
Nombre	Abrir aula para realizar sesión de examen.
Prioridad	Alta
Descripción	Es indispensable abrir una o varias aulas para poder realizar la sesión de examen.
Pasos	▪ Previamente un miembro del tribunal debe iniciar su sesión en la aplicación. ▪ A continuación elegirá la sesión de exámenes que quiere abrir, a través de un desplegable de hora y fecha. ▪ Una vez elegidos ambos, pasará a la siguiente pantalla, en la cual elegirá entre las aulas que se encuentran disponibles, la que desea abrir para realizar la sesión de examen.

Tabla Requisitos Funcionales 12: RF2.5

Identificador	RF2.6
Nombre	Cerrar aula
Prioridad	Alta
Descripción	Es necesario que al finalizar una sesión se cierren las aulas correspondientes a la misma, para su posterior uso en próximas sesiones.
Pasos	▪ Las aulas o el aula utilizadas para una sesión son cerradas automáticamente cuando la sesión a finalizado. Los pasos para el cierre de sesión, están especificados en "*Tabla requisitos funcionales 6*".

Tabla Requisitos Funcionales 13: RF2.6

Identificador	RF2.7
Nombre	Organizar aula
Prioridad	Alta
Descripción	Es fundamental la organización de las aulas que se abren para una sesión, puesto que es una de las características principales de la aplicación, ya que ubica a los estudiantes de forma que no coincidan alumnos de la misma asignatura consecutivamente, dependiendo de la estrategia elegida para su organización.
Pasos	▪ La organización del aula se realiza automáticamente cuando se elige el tipo de estrategia que se quiere seguir en el aula. La disponibilidad de cada posición del aula se almacena en la tabla "Aula_Estrategia".

Tabla Requisitos Funcionales 14: RF2.7

Identificador	RF2.8
Nombre	Visualizar aula
Prioridad	Alta
Descripción	Es necesario para controlar de una forma más visual el estado del aula, para que sea más sencillo ver el estado de los estudiantes que se encuentran en el aula.
Pasos	▪ Un miembro del tribunal debe iniciar sesión en la aplicación. ▪ Seleccionar la sesión que quiere abrir. ▪ Seleccionar el aula donde se va a desarrollar la sesión. ▪ Elegir la estrategia a seguir en la organización del aula, ▪ Una vez lleguemos a la pantalla de entrada de estudiantes, deberemos presionar el botón "Cerrar entrada estudiantes", puesto que solo se puede visualizar el aula cuando hayamos cerrado la entrada de estudiantes. ▪ Después de pulsar dicho botón, la aplicación nos redirige a la pantalla de entrega de exámenes, en la cual podremos visualizar el aula, pulsando "Actualizar Aula".

Tabla Requisitos Funcionales 15: RF2.8

Identificador	RF2.9
Nombre	Consultar datos del aula en la sesión
Prioridad	Media
Descripción	Puede ser útil conocer el número de alumnos que ya han entrado en la sesión, y el número total para llevar un mejor control de los alumnos que faltan por llegar, y prever la apertura de otras aulas, o bien para conocer la cantidad de estudiantes que se han presentado a la sesión.
Pasos	▪ Un miembro del tribunal, debe iniciar una sesión, como hemos indicado anteriormente. ▪ Una vez lleguemos a la pantalla de entrada de estudiantes, se podrá pulsar el botón “Estadisticas”, para consultar datos sobre la sesión. ▪ Este botón también está disponible en la pantalla de entrega de exámenes.

Tabla Requisitos Funcionales 16: RF2.9

Identificador	RF2.10
Nombre	Posicionar al estudiante en el aula
Prioridad	Alta
Descripción	Es necesario guardar la posición donde se ha ubicado al alumno dentro del aula, para poder controlar el estado del mismo y también para conocer los asientos que quedan disponibles en el aula.
Pasos	▪ El estudiante es ubicado en el aula conforme entran en la sesión de examen. Se almacena su DNI en la posición asignada, dependiendo de la estrategia seleccionada, en la tabla “Aula_Estrategia”. De esta forma queda registrada su posición a lo largo de la sesión.

Tabla Requisitos Funcionales 17: RF2.10

5.1.3 Estrategias

En este apartado se encuentran los requisitos necesarios para poder organizar las aulas de examen de acuerdo a un patrón establecido.

Identificador	RF3.1
Nombre	Consultar estrategias
Prioridad	Media
Descripción	Permitir consultar las estrategias que están disponibles para la organización de las aulas
Pasos	

Tabla Requisitos Funcionales 18: RF3.1

Identificador	RF3.2
Nombre	Activar Estrategia
Prioridad	Alta
Descripción	Es imprescindible la elección de una estrategia para la gestión de la sesión de examen, ya que de ella depende la colocación de los alumnos dentro del aula, lo que supone tener un mayor control sobre la gestión de la misma.
Pasos	

Tabla Requisitos Funcionales 19: RF3.2

5.1.4 Estudiantes

En este subsistema se incluyen los requisitos para la gestión de los alumnos en la sesión de exámenes.

Identificador	RF4.1
Nombre	Verificar estudiante
Prioridad	Alta
Descripción	Es necesario comprobar la identificación del alumno, esto se realiza mediante la lectura de su carnet.
Pasos	▪ Esta comprobación se realiza en la pantalla de entrada de estudiantes, cuando se presiona el botón “Comprobar”, después de haber escaneado el carnet del estudiante.

Tabla Requisitos Funcionales 20: RF4.1

Identificador	RF4.2
Nombre	Comprobar asignaturas del estudiante
Prioridad	Alta
Descripción	Una vez se comprueba la identificación del estudiante, se coteja si este alumno tiene alguna asignatura que corresponda a la sesión en cuestión.
Pasos	▪ Después de verificar el carnet del estudiante como hemos indicado en la tabla anterior, se pasa a comprobar si el alumno está matriculado en alguna de las asignaturas que se realizan en esa sesión.

Tabla Requisitos Funcionales 21: RF4.2

Identificador	RF4.3
Nombre	Control tiempo de examen en la sesión
Prioridad	Alta
Descripción	Este requisito nos permite controlar si el alumno ha agotado su tiempo para realizar el examen, es decir, si entrega el examen dentro o fuera de tiempo.
Pasos	▪ Este control se realiza cuando el estudiante entrega el examen.

Tabla Requisitos Funcionales 22: RF4.3

5.1.5 Convocatoria

En este subsistema vamos a definir los requisitos necesarios para gestionar los datos de la convocatoria para cada una de las sesiones.

Identificador	RF5.1
Nombre	Insertar nueva convocatoria
Prioridad	Alta
Descripción	Se inserta una nueva convocatoria cada vez que se posiciona un nuevo alumno en el aula.
Pasos	▪ Tras colocar al alumno en una posición del aula se guardan automáticamente los datos de dicho estudiante en la tabla "Convocatoria". ▪ Esos datos se vuelven a modificar cuando el estudiante entrega el examen, insertando la hora de entrega y el estado final del estudiante

Tabla Requisitos Funcionales23: RF5.1

Identificador	RF5.1
Nombre	Modificar datos de la convocatoria
Prioridad	Media
Descripción	Puede ser necesario modificar algún dato de la convocatoria en el caso que sea preciso.
Pasos	▪ Un miembro del tribunal debe iniciar sesión en la aplicación. ▪ Debe elegir la sesión que quiere abrir seleccionando la fecha y la hora ▪ A continuación seleccionará el aula y la estrategia para posicionar a los estudiantes. ▪ Una vez en la pantalla de entrada de estudiantes, por cada alumno que entre a la sesión se insertan automáticamente los datos en la tabla "Convocatoria".

Tabla Requisitos Funcionales 24: RF5.2

Identificador	RF5.3
Nombre	Visualizar datos de la convocatoria actual
Prioridad	Alta
Descripción	Se visualizan los datos de la convocatoria actual al finalizar la sesión, a modo resumen de la convocatoria realizada.
Pasos	▪ Un miembro del tribunal debe iniciar sesión en la convocatoria. ▪ A continuación elige la fecha y hora de la sesión. ▪ Selecciona el aula y la estrategia. ▪ Tras cerrar la entrada de estudiantes y después de que todos los estudiantes hayan entregado sus exámenes se debe pulsar el botón “Cerrar sesión” para visualizar los datos de la convocatoria.

Tabla Requisitos Funcionales 25: RF5.3

5.1.6 Incidencias

En este subsistema definiremos las tareas relacionadas con las incidencias que pueden ocurrir en el desarrollo de la sesión.

Identificador	RF6.1
Nombre	Añadir nueva incidencia
Prioridad	Media
Descripción	Se inserta una nueva incidencia a la sesión de examen en curso.
Pasos	▪ Para crear una nueva incidencia deberemos iniciar la aplicación y llegar hasta la pantalla de acceso a estudiantes, como hemos indicado en tablas anteriores. ▪ A continuación debemos presionar el botón “Incidencias” y seleccionar el enlace “Create New”, el cual nos llevará a un formulario donde tendremos que rellenar los campos requeridos para completar la incidencia. ▪ Una vez escritos todos los campos presionaremos el botón “Create”, que nos llevará a la lista de incidencias.

Tabla Requisitos Funcionales 26: RF6.1

5.2 Requisitos no funcionales

En este punto pasamos a definir los requisitos no funcionales de la aplicación, los cuales hacen referencia a características que no están relacionados con la funcionalidad de la aplicación.

Identificador	RNF1
Tipo	Recursos necesarios
Prioridad	Alta
Descripción	La aplicación podrá ser ejecutada sobre cualquier pc cuyo sistema operativo sea Windows, se disponga de una impresora y de un lector de código de barras.

Tabla Requisitos No Funcionales 1: RNF1

Identificador	RNF2
Tipo	Usabilidad
Prioridad	Alta
Descripción	La interfaz de la aplicación debe ser simple y sencilla, haciendo que el uso de ésta sea intuitivo y no requiera un estudio detallado previamente.

Tabla Requisitos No Funcionales 2:RNF2

Identificador	RNF3
Tipo	Seguridad del sistema
Prioridad	Alta
Descripción	Solo se podrá utilizar la aplicación si previamente uno de los miembros del tribunal se ha autentificado con su nombre y contraseña.

Tabla Requisitos No Funcionales 3:RNF3

Identificador	RNF4
Tipo	Eficiencia
Prioridad	Alta
Descripción	La aplicación se desarrollará de forma escalable, para que en un futuro pueda ser posible aumentar la funcionalidad aprovechando el código ya existente.

Tabla Requisitos no Funcionales 4:RNF4

Identificador	RNF5
Tipo	Eficiencia
Prioridad	Alta
Descripción	Se realizará un diseño modular, para facilitar la unión en un futuro de nueva funcionalidad.

Tabla Requisitos no Funcionales 5: RNF5

Identificador	RNF6
Tipo	Implementación
Prioridad	Alta
Descripción	La aplicación debe presentar un fácil mantenimiento, garantizado por: • Comentarios en el código. • Una correcta documentación • Uso de patrones de diseño

Tabla Requisitos no Funcionales 6: RNF6

Identificador	RNF7
Tipo	Rendimiento
Prioridad	Alta
Descripción	La aplicación debe tener un tiempo de respuesta mínimo en las consultas a la base de datos, en ningún caso se superarán los 5 segundos.

Tabla Requisitos no Funcionales 7:RNF7

6 Diseño

En este capítulo vamos a describir el diseño de la aplicación, partiendo del catálogo de requisitos definido en el apartado anterior, como se ha desarrollado su implementación, y las decisiones tomadas para realizarlo.

6.1 Esquema general

El esquema general de la aplicación lo hemos dividido en tres secciones para una mayor comprensión:

1. Antes de iniciar la sesión de exámenes:

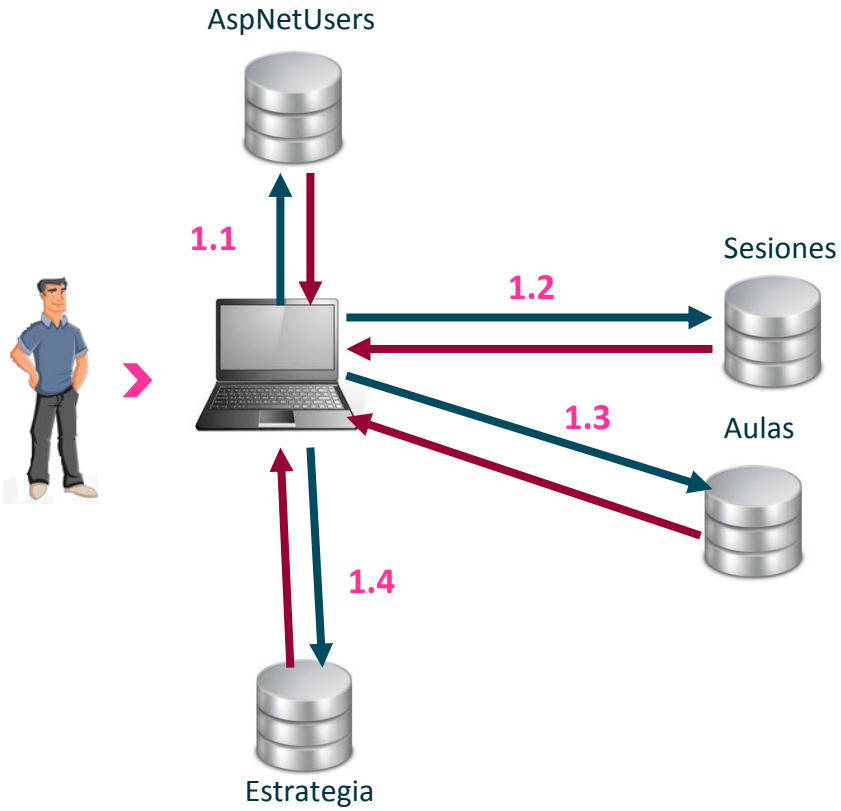


Figura 1: Previo a la entrada de estudiantes

A continuación explicaremos brevemente cada uno de los puntos:

1.1. Un miembro del tribunal se autentifica en la aplicación, para comenzar con el proceso de la sesión. Comprobando el nombre introducido y contraseña en la tabla "AspNetUsers".

1.2. Una vez "logueado", se podrá abrir la sesión correspondiente, estableciendo su estado como abierta.

1.3. A continuación, se elige el aula donde se desee que se realice la sesión seleccionada anteriormente, pudiendo escoger entre aquellas que se encuentren cerradas en la tabla "Aulas".

1.4. Cuando ya está abierta el aula, se procede a elegir la estrategia que se debe seguir para la ubicación de los estudiantes en la misma.

A partir de este momento, ya puede comenzar la entrada de estudiantes a la sesión de exámenes.

2. Durante la sesión de exámenes:

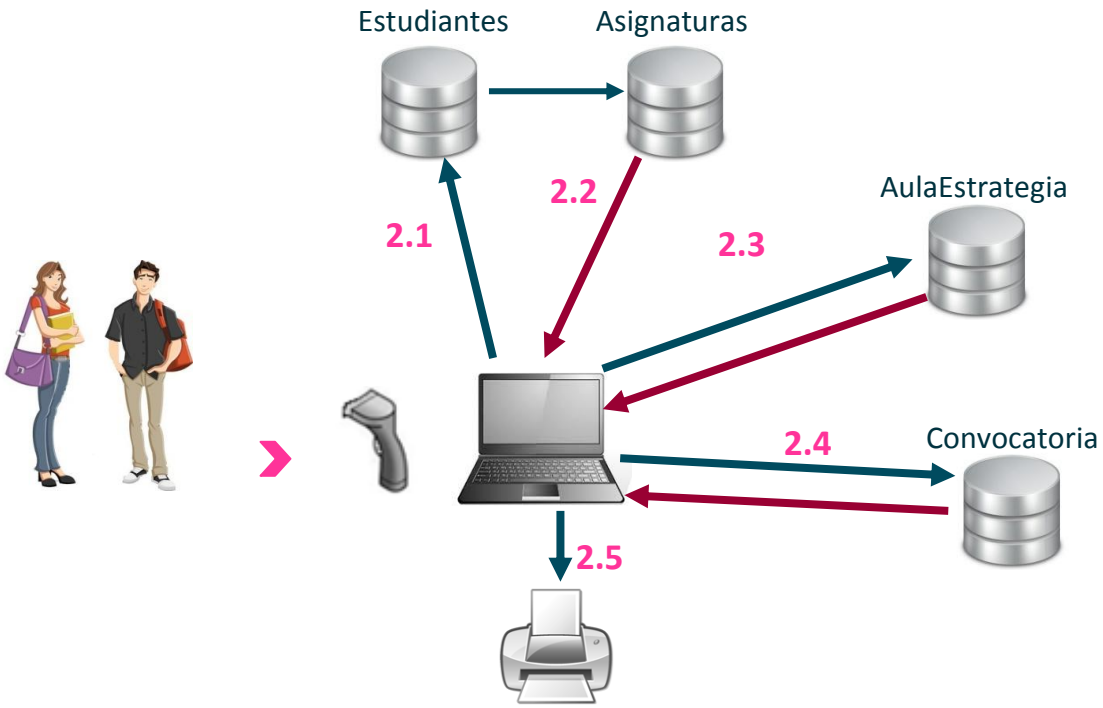


Figura 2: Entrada estudiantes a la sesión

2.1. Cuando llegan los estudiantes al aula se lee su DNI con un lector de códigos de barra, éste se busca en la tabla “Estudiantes”.

2.2. Si el DNI es correcto, se buscaran las asignaturas en la tabla “Asignaturas” de las que se puede examinar dicho estudiante en esa sesión.

2.3. Seguidamente, se ubica al alumno dentro del aula siguiendo la estrategia elegida anteriormente, actualizándose la tabla “AulaEstrategia”

2.4. Una vez situado, se guarda la información del estudiante en la tabla “Convocatoria”, para su posterior actualización.

2.5. Por último se procede a imprimir el examen del alumno.

3. Al finalizar la sesión

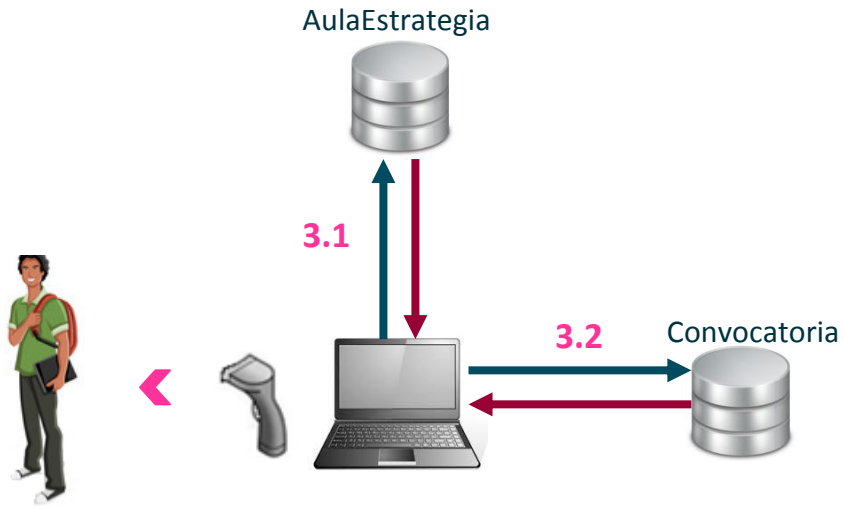


Figura 3: Salida de estudiantes de la sesión

3.1. El alumno entrega el examen se escanea su código de barras y se actualiza el estado de su posición en el aula en la tabla “AulaEstrategia”.

3.2. Una vez se ha actualizado, se guardan los datos de entrega del alumno en la tabla “Convocatoria”, actualizando de igual modo el estado del estudiante.

6.2 Diseño técnico de la aplicación

En este capítulo abordaremos los aspectos técnicos del diseño de la aplicación, contando con los requisitos que hemos comentado anteriormente.

Se analizará el diseño de la base de datos, detallando la utilidad de cada una de las tablas en la aplicación.

6.2.1 Base de datos

Para el diseño de los datos del proyecto "Gestión de aulas de examen", se ha implementado la base de datos **ValijaUam** con **SQL Server**, en la cual se representan las distintas tablas, que contienen la información para dicha aplicación.

La elección de SQL Server como sistema para la gestión de bases de datos, se debe a que mantiene una mayor concordancia con la tecnología de Microsoft utilizada para el proyecto. El empleo de SQL Server en esta aplicación no excluye el uso conjunto de la aplicación con otras (Oracle, PostgreSQL, mySQL…) ya que el framework es compatible con todas las bases de datos más comunes.

A continuación describimos cada una de las tablas que componen la base de datos:

6.2.1.1 Asignatura

Nombre	Asignatura
Descripción	Tabla donde se almacenan los datos de las asignaturas cuyo examen se realiza en alguna de las sesiones.
Atributos	▪ [id_asignatura] INT ▪ [id_sesion] INT ▪ [nombre] NVARCHAR (50) ▪ [carrera] NVARCHAR (50) ▪ [duracion] INT ▪ [material] NVARCHAR (50) ▪ [alumnos_matriculados] INT ▪ [examen] NVARCHAR (100)

Tabla Datos 1: Asignatura

En la tabla asignatura, se encuentran almacenados los datos necesarios para definir cada una de las asignaturas, así como la duración y los materiales permitidos para el examen de dicha materia. También se guardan el número de alumnos matriculados, para poder hacer un recuento, sobre cuantas posiciones serán necesarias en las distintas aulas donde se realice el examen. Por otro lado, en esta tabla, se guarda la ruta en la que se encuentra el examen de cada asignatura, dicha ruta se almacena en el atributo "examen".

El tipo de datos "*nvarchar*", se ha utilizado para los atributos: carrera, material y examen, ya que la longitud de las distintas cadenas que se introducen para dichas columnas son variables. Se ha escogido el tipo "*int*" para id_sesion, curso, alumnos_matriculados y duración, éste último hace referencia a las horas que dura el examen en cuestión. El identificador de asignatura es un tipo entero que se va autoincrementando en una unidad, a medida que introducimos una nueva entrada para esta tabla.

Clave primaria: "id_asignatura", el identificador de la asignatura.

Claves foráneas: "id_sesion", hace referencia al identificador de la tabla "Sesion","id_sesion".

6.2.1.2 Asignatura_Estudiante

Nombre	Asignatura_Estudiante
Descripción	Tabla donde se almacenan las asignaturas de las que está matriculado cada estudiante
Atributos	▪ [dni_estudiante] VARCHAR (9) ▪ [id_asignatura] INT

Tabla Datos 2: Asignatura_Estudiante

Esta tabla guarda la relación entre un Estudiante y sus Asignaturas matriculadas, e inversamente, una Asignatura con los Estudiantes que están matriculados en ésta.

Para ello se almacena el DNI del estudiante con el correspondiente identificador de la asignatura "id_asignatura", tantas veces como asignaturas esté matriculado el estudiante.

Clave primaria: "dni_estudiante" junto con "id_asignatura". Ya que son identificadores de las tablas que representan "Estudiante" y "Asignatura".

Claves foráneas: "id_asignatura", hace referencia al "id_asignatura" de la tabla "Asignatura", y "dni_estudiante", hace referencia a la tabla "Estudiante".

6.2.1.3 Asignaturas_Reservadas

Nombre	Asignaturas_Reservadas
Descripción	Tabla donde se almacenan las asignaturas que quedan reservadas para ser realizadas en otra sesión
Atributos	▪ [id_reserva] INT IDENTITY (1, 1) ▪ [dni_estudiante] VARCHAR (9) ▪ [id_asignatura] INT

Tabla Datos 3: Asignaturas_Reservadas

En la tabla Asignatuas_Reservadas, se van almacenando las asignaturas cuyo examen, coinciden con otra materia de las que está matriculado el alumno, de esta forma esas asignaturas quedan reservadas para hacerlas en otra sesión. Se almacenan guardando el DNI del estudiante "dni_estudiante", junto con el identificador de la asignatura que le coincide con la sesión de otro examen, "id_asignatura".

Clave primaria: "id_reserva", el identificador de la reserva.

6.2.1.4 Aula_Estrategia

Nombre	Aula_Estrategia
Descripción	Tabla donde se almacenan las posiciones del aula escogida para realizar los exámenes de la sesión
Atributos	▪ [posicion] INT IDENTITY (1, 1) ▪ [id_aula] INT ▪ [pos_x] INT ▪ [pos_y] INT ▪ [permitida] BIT ▪ [id_asignatura] INT ▪ [estado] NVARCHAR (50) ▪ [dni] VARCHAR (9)

Tabla Datos 4: Aula_Estrategia

Se almacenan las posiciones del aula actual dónde se está realizando la sesión, en la que están entrando los estudiantes para realizar el examen, de forma que cada uno de los asientos están marcados mediante el atributo 'permitida', dependiendo de la estrategia seleccionada. Esta tabla se utiliza para poder representar el plano del aula, junto con la tabla Convocatoria, que se explicara más adelante.

Los tipos de datos utilizados son "*int*" enteros para representar las posiciones y el identificador del aula en cuestión y "*bit*" para indicar si está permitida o no.

El atributo "id_asignatura", se utiliza para indicar, en el caso que la posición esté permitida, la asignatura de la que se examina el estudiante que ocupa dicha posición, identificado mediante el atributo "dni", para evitar que estudiantes que realizan la misma materia se sitúen en las posiciones de alrededor.

También hemos creado el atributo "estado", el cual es fundamental para la representación del aula, puesto que indica que posiciones están ocupadas, de entre las que son permitidas por la estrategia.

Clave primaria: "posicion", identificador de la tabla "Aula_Estrategia"**.**

6.2.1.5 Aulas

Nombre	Aulas
Descripción	Tabla donde se almacenan los datos de las distintas aulas de las que se dispone para realizar las sesiones
Atributos	▪ [id_aula] INT IDENTITY (1, 1) ▪ [n_filas] INT ▪ [n_columnas] INT ▪ [id_sesion] INT ▪ [abierta] BIT ▪ [capacidad] INT

Tabla Datos 5: Aulas

En esta tabla, se almacena la información referente a las aulas donde se realizan los distintos exámenes en las sesiones, aquí se encuentran el numero de filas y columnas para poder determinar los asientos que serán ocupados dependiendo de la estrategia que haya seleccionado el tribunal para cada aula. Se dispone de la capacidad de cada una de ellas,

para saber de antemano cuantas aulas serán necesarias para albergar los exámenes de distintas asignaturas. También tenemos el atributo “abierta” para indicar si un aula está ya abierta para una sesión.

El tipo de dato seleccionado para el identificador del aula es como en tablas anteriores un entero que se va autoincrementando a la hora que introducimos una nueva entrada a la tabla, para el numero de filas, columnas el identificador de sesión y la capacidad del aula usamos “int”, puesto que serán números enteros, y el atributo reservada se representa con “bit”, un tipo booleano.

El atributo “id_sesion”, es necesario para saber que aulas se están utilizando para la misma sesión

Clave primaria: ”id_aula”, identificador del aula.

Claves foráneas: “id_sesion”, hace referencia al identificador de la tabla “Sesion”,”id_sesion”.

6.2.1.6 Convocatoria

Nombre	Convocatoria
Descripción	Tabla donde se guardan los datos de cada alumno con respecto a la sesión a la que se ha presentado
Atributos	▪ [id_convocatoria] INT IDENTITY (1, 1) ▪ [dni_estudiante] VARCHAR (9) ▪ [id_aula] INT ▪ [pos_x] INT ▪ [pos_y] INT ▪ [sesion] INT ▪ [asignatura] INT ▪ [hora_ini] TIME (7) ▪ [hora_fin] TIME (7) ▪ [estado] NVARCHAR (40)

Tabla Datos 6: Convocatoria

Esta tabla recoge la información del estudiante que acaba de entrar en la sesión de examen, en ella se guardan los atributos indicados en la tabla, de forma que el plano del aula se podrá actualizar, para ver las posiciones que han sido ocupadas y el estado en el

que se encuentra el alumno (dentro de tiempo o fuera de tiempo), dependiendo de la hora de inicio y la hora fin del examen.

El tipo de dato utilizado para el identificador de la tabla id_convocatoria es "int identity" un valor entero que se va autoincrementando, a medida que los alumnos entran a la sesión, las horas de inicio y fin del examen son de tipo "time", para poder guardarlas con el formato adecuado, las cuales indican la hora de entrada del estudiante y la hora a la que entregó el examen.

Clave primaria: "id_convocatoria", identificador de la convocatoria.

6.2.1.7 Estrategia

Nombre	Estrategia
Descripción	Tabla que contiene los tipos de estrategia que se pueden escoger para organizar el aula
Atributos	▪ [nombre] NVARCHAR (30) ▪ [filas] BIT ▪ [columnas] BIT

Tabla Datos 7: Estrategia

En esta tabla se encuentran los datos de las estrategias que se pueden elegir para la distribución de los alumnos en las aulas.

Los atributos filas y columnas, indican si se quitan filas o columnas, dependiendo de la estrategia elegida, es por eso que son de tipo "bit".

El nombre es de tipo "nvarchar", puesto que los nombres de las estrategias son de distinta longitud.

Clave primaria: "nombre", nombre de las estrategias.

6.2.1.8 Estudiante

Nombre	Estudiante
Descripción	Tabla que contiene la información relacionada con los estudiantes que están matriculados en alguna de las asignaturas que se realizan en las sesiones
Atributos	▪ [dni] VARCHAR (9) ▪ [nombre] NVARCHAR (20) ▪ [apellido] NVARCHAR (30) ▪ [curso] INT ▪ [carrera] NVARCHAR (50) ▪ [id_reserva] INT ▪ [id_convocatoria] INT

Tabla Datos 8: Estudiante

En la tabla Estudiante, se encuentran los datos relativos a un alumno, como se muestra en la tabla anterior. El tipo de dato escogido para el DNI es "*varchar*" que tendrá una longitud fija de 9 caracteres, puesto que todos los DNI tienen esta longitud, en cambio para nombre, apellido y carrera el tipo seleccionado ha sido "*nvarchar*", ya que puede ir variando la longitud de estas columnas.

El atributo "id_reserva" almacena el identificador de la asignatura que el estudiante haya decidido realizar en otra sesión, debido a la coincidencia de dos asignaturas en la misma sesión de examen. En "id_convocatoria", se almacena el identificador de la convocatoria actual en la que se encuentra el estudiante, de este modo se puede consultar la tabla de estudiantes para verificar que concuerdan los datos entre la convocatoria y el estudiante.

Clave primaria: "dni", DNI del estudiante.

Claves foráneas: "id_convocatoria", hace referencia a "id_convocatoria" de la tabla "Convocatoria" y "id_reserva", hace referencia a "id_reserva" de la tabla "Asignaturas_Reservadas".

6.2.1.9 Incidencias

Nombre	Incidencias
Descripción	Tabla donde se almacenarán aquellas incidencias que puedan ocurrir a lo largo de una sesión.
Atributos	▪ [id_incidencia] INT IDENTITY (1, 1) ▪ [nombre] NVARCHAR (50) ▪ [sesion] INT ▪ [tipo] NVARCHAR (50) ▪ [implicados] NVARCHAR (MAX) ▪ [descripcion] NVARCHAR (MAX)

Tabla de Datos 9: Incidencias

En la tabla Incidencias, se almacenan aquellos "problemas" que puedan surgir a lo largo del desarrollo de la sesión, de forma que el tribunal en cualquier momento pueda escribir una nueva incidencia, para ello será necesario que se rellenen todos los atributos de la tabla.

El atributo "nombre", es necesario para saber que miembro del tribunal creo la incidencia en caso de consulta, "sesion", fundamental para conocer en que sesión de exámenes se produjo la incidencia, "tipo", este atributo se debe completar para conocer sin entrar en detalle si se trata de copia, material no permitido, etc o cualquier otro tipo, en el atributo "implicados" se deberá escribir el nombre de los estudiantes que intervienen en dicha incidencia, y por ultimo "descripción", donde se narrará con detalle lo sucedido.

Clave primaria: "id_incidencia", identificador de las incidencias.

Claves foráneas: "sesion", hace referencia a "id_sesion" de la tabla "Sesiones".

6.2.1.10 Sesiones

Nombre	Sesiones
Descripción	Tabla donde se encuentran los datos correspondientes a las sesiones que se realizarán
Atributos	▪ [id_sesion] INT IDENTITY (1, 1) ▪ [fecha_sesion] DATE ▪ [hora_sesion] TIME (7) ▪ [estado] BIT

Tabla datos 10: Sesiones

En la tabla Sesiones, se almacena la información correspondiente de las sesiones de exámenes, como es la fecha en la que se realiza, la hora en la que se inicia y el estado de la sesión, es decir, abierta o cerrada.

Para cada una de las sesiones tendremos dos horas distintas de inicio de sesión (mañana y tarde).

Los tipos de datos escogidos para estos atributos son: "*int identity*", para el identificador de sesión, este valor se va autoincrementando en una unidad, a medida que insertamos una nueva sesión en la base de datos, para la fecha y hora se usan "*date*" y *"time"* respectivamente, para usar el formato de fecha "dd/mm/aaaa" y hora "hh/mm/ss", a la hora de crear y abrir la sesión.

Clave primaria: "id_sesion", identificador de las sesiones.

6.2.1.11 Tribunal (AspNetUsers)

Nombre	Tribunal
Descripción	Tabla donde se encuentran los datos correspondientes a las sesiones que se realizarán
Atributos	▪ [Id] NVARCHAR (128) ▪ [UserName] NVARCHAR (MAX) ▪ [PasswordHash] NVARCHAR (MAX) ▪ [SecurityStamp] NVARCHAR (MAX) ▪ [Discriminator] NVARCHAR (128)

Tabla Datos 11: Tribunal

Es una tabla propia de Asp.Net. En esta tabla se almacenan los datos de los miembros del tribunal, quienes deberán utilizar su “UserName” y “PasswordHash”, para comenzar a usar la aplicación.

Clave primaria:”Id”, identificador de la tabla “AspNetUsers”.

6.2.2 Diagrama relacional

En este apartado explicaremos las relaciones que existen entre las distintas tablas de la base de datos.

Un estudiante puede estar matriculado de muchas asignaturas, y una asignatura la pueden cursar muchos estudiantes

Una convocatoria puede contener a muchos estudiantes en una sesión, pero un estudiante solo tiene una convocatoria en una sesión.

Varios estudiantes pueden tener una asignatura reservada en la misma sesión, del mismo modo una misma asignatura puede estar reservada por muchos estudiantes en la misma sesión.

Una asignatura solo puede pertenecer a una sesión, poniendo como excepción aquellas que pasan a ser asignaturas reservadas. Mientras que una sesión tiene varias asignaturas.

Una sesión puede tener varias incidencias. Una misma incidencia solo puede pertenecer a la sesión en la que se produjo.

Una sesión se puede desarrollar en varias aulas, mientras que en un aula solo se puede desarrollar una sesión en un tiempo determinado.

A continuación mostramos el **diagrama relacional** de la aplicación:

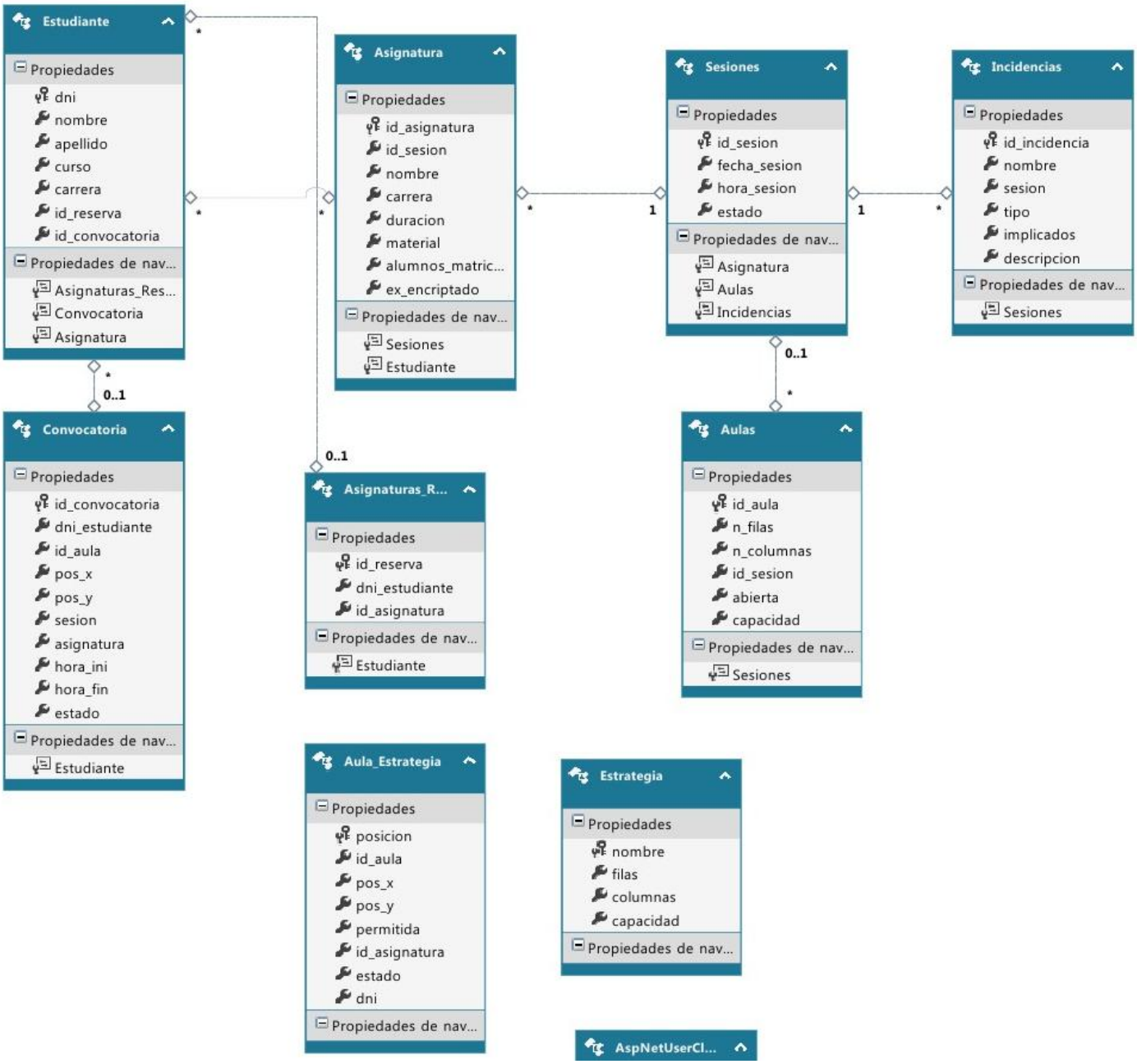


Figura 4 : Diagrama Relacional

6.3 Descripción de las fases de la aplicación

A continuación se procede a desarrollar la relación entre las distintas tablas de la base de datos **BD_VALIJA.**

El proceso que sigue la aplicación "*Gestión de Aulas de Examen"* se puede desglosar en tres partes:

6.3.1 Previo a la sesión

Para comenzar a usar la aplicación, es necesario que uno de los miembros del Tribunal se autentifique, introduciendo su usuario y contraseña. Los datos insertados se comprueban en la tabla *"AspNetUsers"*, si se encuentran en la base de datos y ha hecho el login correcto, pasarán a elegir la sesión que quieren abrir, seleccionando la fecha y hora de la misma.

Una vez seleccionados se almacena en una variable de Session del proyecto llamada "id_sesion", para poder acceder en cualquier momento al identificador de la sesión que hemos seleccionado.

A continuación, se procede a escoger el aula que queremos abrir para la realización de la sesión, para ello se muestran aquellas que se encuentran cerradas, ya que las que están abiertas están siendo utilizadas para la misma sesión.

El tipo de estrategia es la siguiente característica a seleccionar, para ello debemos escoger entre los dos tipos existentes "Quita filas" o "Quita columnas".

Con la estrategia elegida, se procede a insertar los datos en la tabla *"Aula_Estrategia"*, con el *"id_aula"*, del aula seleccionada, y utilizando el *"n_filas"* y *"n_columnas"*, se rellenan cada una de las posiciones con TRUE/FALSE, dependiendo si la estrategia permite ocupar ese asiento o no.

6.3.2 Durante la sesión

Durante la sesión de exámenes, se procede a la lectura del carnet del estudiante, para ello se busca el *"dni"* del estudiante en la tabla *"Estudiante"*, y a continuación, se obtienen sus asignaturas y se comprueban si esas asignaturas tienen relacionada la sesión actual que

se está desarrollando, si pertenece a la sesión, en el caso de que tenga más de una asignatura para evaluación, el estudiante deberá elegir de cual quiere examinarse. La asignatura que el estudiante no ha seleccionado se guarda en la tabla *"Asignaturas_Reservadas"* con el *"id_asignatura"* y el *"DNI"* del *"Estudiante"*.

Una vez seleccionada la asignatura de la cual va a realizar el examen, se busca la posición en la cual debe situarse el alumno dentro del aula, para ello se consulta que posición en la tabla *"Aula_Estrategia"*, se encuentra *"permitida"* y a continuación se comprueba que los alumnos que estén sentados a sus lados no se estén evaluando de la misma asignatura. Cuando ya se tiene la posición del estudiante, se procede a insertar los datos en la tabla *"Convocatoria".* Acto seguido se imprime el examen del estudiante.

6.3.3 Al finalizar la sesión

Para finalizar la sesión de examen es necesario que todos los alumnos de la misma hayan entregado el examen correspondiente.

La entrega se realiza, escaneando el código que los alumnos pegan en su examen, al ser escaneado, se muestra la información del estudiante y al mismo tiempo se guarda el estado final en la tabla "Convocatoria".

Una vez se han entregado todos los exámenes, se pasa a cerrar el aula donde se ha realizado dicha sesión, para que pueda ser utilizada en sesiones posteriores.

Por último al finalizar la sesión aparece por pantalla un resumen con los datos de la de la tabla *"Convocatoria"*, relacionados con la sesión que se acaba de realizar.

6.4 Algoritmos

Se han implementado distintos algoritmos para clasificar las posiciones permitidas y no permitidas según la estrategia escogida por el tribunal.

- Estrategia "**Quita columnas**"

  Recorremos todas las columnas, si el numero de columna es par, recorremos cada una de sus posiciones estableciendo dichas posiciones como "no permitidas". Si el número de columna es impar, hacemos el mismo proceso pero estableciendo dichas posiciones como "permitidas".

La implementación del algoritmo la podemos ver en la siguiente imagen:

```
for (i = 1; i <= aula.First().n_columnas; i++)
{
    if (i % 2 == 0)
        quitar = true;
    else
        quitar = false;
    for (j = 1; j <= aula.First().n_filas; j++)
    {
        ae.id_aula = aula.First().id_aula;
        ae.pos_y = j;
        ae.pos_x = i;
        if (quitar == true)
        {
            ae.permitida = false;
            ae.estado="np";
        }
        else
        {
            ae.permitida = true;
            ae.estado = "p";
        }


        if (ModelState.IsValid)
        {
            db.Aula_Estrategia.Add(ae);
            db.SaveChanges();
        }

    }
```

Figura 5: Algoritmo columnas

- Estrategia "**Quita filas**"

  Para realizar la clasificación de asientos permitidos con la estrategia de eliminar filas del aula, seguimos el mismo procedimiento que para la estrategia anterior.

  Recorremos todas las filas, si el numero de fila es par, recorremos cada una de sus posiciones estableciendo dichas posiciones como "no permitidas". Si el número de fila es impar, hacemos el mismo proceso pero estableciendo dichas posiciones como "permitidas".

  La implementación del algoritmo la podemos ver en la siguiente imagen:

```
for (i = 1; i <= aula.First().n_filas; i++)
{
    if (i % 2 == 0)
        quitar = false;
    else
        quitar = true;
    for (j = 1; j <= aula.First().n_columnas; j++)
    {
        ae.id_aula = aula.First().id_aula;
        ae.pos_y = i;
        ae.pos_x = j;
        if (quitar == true)
        {
            ae.permitida = false;
            ae.estado = "np";
        }
        else
        {
            ae.permitida = true;
            ae.estado = "p";
        }
        if (ModelState.IsValid)
        {
            db.Aula_Estrategia.Add(ae);
            db.SaveChanges();
        }

    }
}
```

Figura 6: Algoritmo filas

Otro de los algoritmos implementados, se utiliza para situar a los estudiantes dentro del aula, evitando que alumnos con la misma asignatura se sienten consecutivamente.

Comprobar que el alumno de la posición anterior no tenga la misma asignatura, igualmente hay que comprobar la posición siguiente, ya que puede que este ocupada por otro alumno.

En el caso que no tengan la misma asignatura el alumno se sitúa en dicha posición.

Si tienen la misma asignatura, pasamos a la siguiente posición permitida por la estrategia que se encuentre libre.

Estos pasos se realizan consecutivamente hasta que el aula está completa.

7 Implementación

7.1 Descripción del patrón

En este capítulo pasamos a hablar del **MVC Framework**, que hemos señalado anteriormente, éste es un conjunto de plantillas para "VisualStudio" y bibliotecas en el namespace "System.Web.Mvc" que permite crear aplicaciones ASP.Net basadas en el patrón de diseño Modelo-Vista-Controlador(MVC).

El modelo arquitectónico Modelo-Vista-Controlador (MVC) separa una aplicación en tres componentes principales: el modelo, la vista y el controlador. El marco de ASP.NET MVC proporciona una alternativa al modelo de formularios Web Forms de ASP.NET para crear aplicaciones web. El marco de ASP.NET MVC es un marco de presentación de poca complejidad y fácil de comprobar que (como las aplicaciones basadas en formularios Web Forms) se integra con las características de ASP.NET existentes, tales como páginas maestras y la autenticación basada en pertenencia. El marco de MVC se define en el ensamblado System.Web.Mvc.

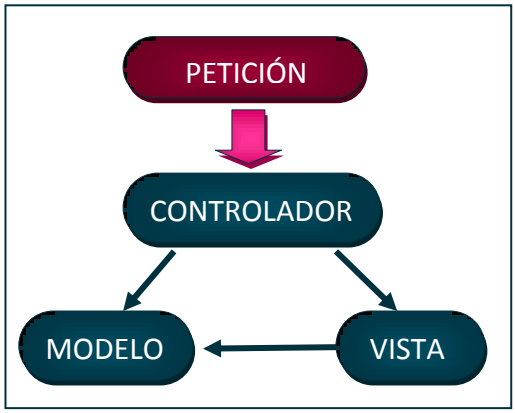


Figura 7: Funcionamiento MVC

Los proyectos basados en el MVC se crean partiendo de la plantilla de proyecto "ASP.NET MVC Web Application" en VisualStudio.

Al crear el nuevo proyecto se crearan las carpetas /Controllers, /Models, /Views, estas carpetas servirán para almacenar las clases del Controlador, Modelo, y Vista respectivamente.

A continuación se detallan las características de cada una de estas capas de la arquitectura:

7.1.1 Modelo

Los objetos de modelo son las partes de la aplicación que implementan la lógica del dominio de datos de la aplicación. A menudo, los objetos de modelo recuperan y almacenan el estado del modelo en una base de datos. Por ejemplo, un objeto Product podría recuperar información de una base de datos, trabajar con ella y, a continuación, escribir la información actualizada en una tabla Productos de una base de datos de SQL Server.

7.1.2 Vista

Las vistas son los componentes que muestra la interfaz de usuario de la aplicación. Esta interfaz de usuario se crea a partir de los datos de modelo. Para ello la versión de "Visual Studio 2013", cuenta con Twitter Bootstrap como framework de interfaz de usuario. Bootstrap es una colección de varios elementos web personalizables y funciones completamente empaquetado en una sola herramienta, son una combinación de HTML , JavaScript y CSS para el diseño de formularios, botones,etc.

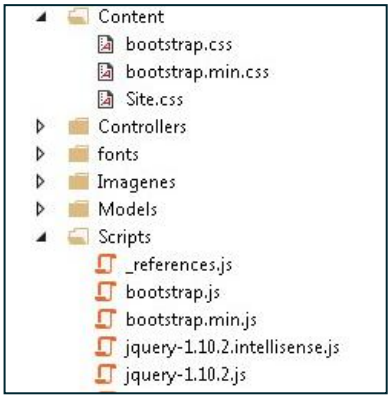


Figura 8: Bootstrap

7.1.3 Controlador

Los controladores son los componentes que controlan la interacción del usuario, trabajan con el modelo y por último seleccionan una vista para representar la interfaz de usuario. En una aplicación MVC, la vista solo muestra información; el controlador administra y responde a los datos proporcionados por el usuario y su interacción. Por ejemplo, el controlador administra los valores de la cadena de consulta y pasa estos valores al modelo, que a su vez podría usarlos para consultar la base de datos.

El framework MVC permite configurar el mapeo de URL's a distintas clases del controlador. Por defecto y sin tener que realizar ninguna configuración las URL's tipo /xxxxx/ se asocian con la clase xxxxController. Podemos ver su funcionamiento en la siguiente imagen:

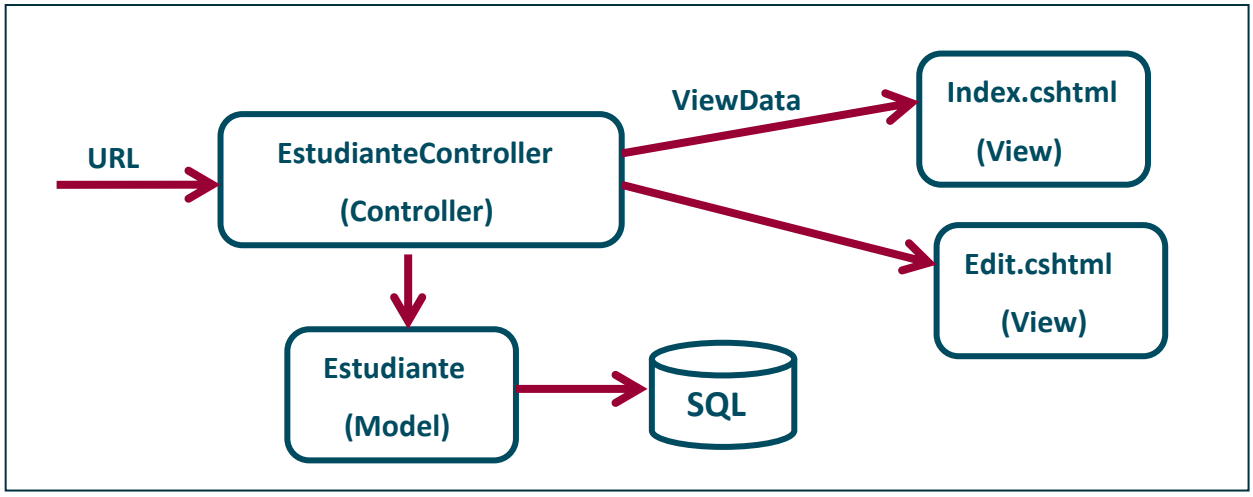


Figura 9: Petición URL

La clase controladora es responsable de ejecutar una determinada lógica en función de las acciones que se producen en la aplicación.

Convierte las URL's en llamadas a métodos (acción del controlador) de la clase controladora, por ejemplo la Url: "/Estudiantes/Index" produce una invocación del método "Index" de la clase controladora "Estudiantes". Además transforma automáticamente los parámetros de la querystring en los parámetros de entrada de los métodos acciones de la clase controladora. Una vez que el Controlador esta "controlando" la acción de la aplicación debe obtener o entregar el dato a la clase Modelo y visualizar la Vista.

7.1.3.1 Scaffolding

Scaffolding (generación de código), hace referencia a los elementos de datos dinámicos que generan automáticamente páginas web para cada tabla de una base de datos. Estas páginas web generadas automáticamente proporcionan operaciones de creación, lectura, actualización y eliminación (CRUD) para cada tabla.

Las plantillas que genera son: “Details.aspx”, “Edit.aspx”, “Insert.aspx”,” List.aspx”.

Veamos un ejemplo, utilizando el modelo correspondiente a “Sesiones”:

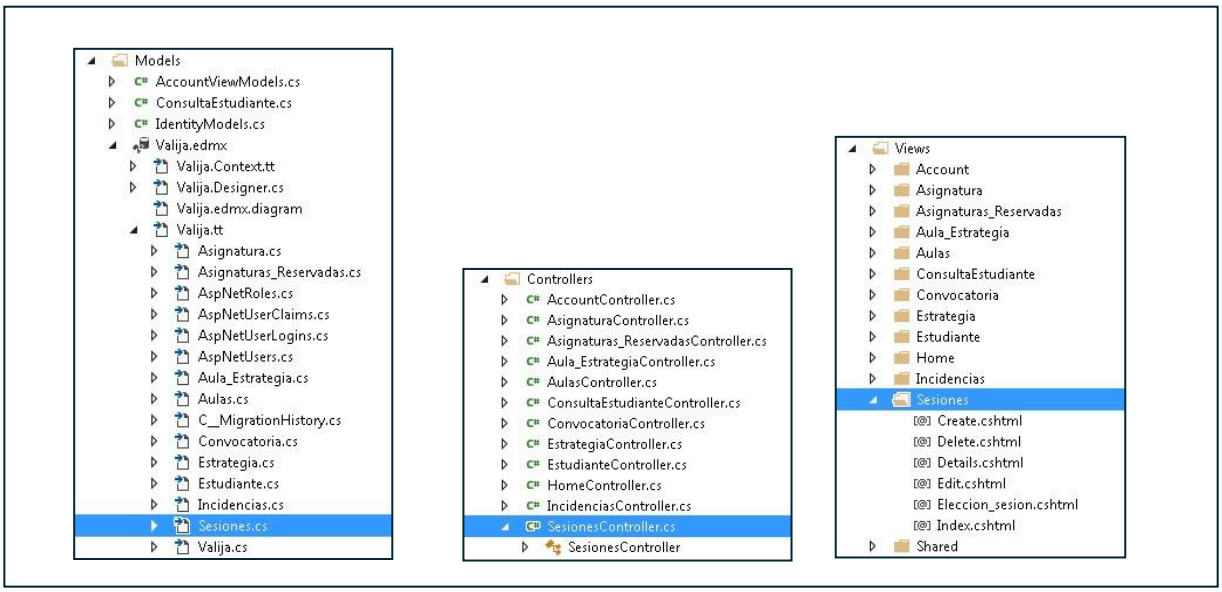


Figura 10: Scafolding

8 Pruebas y resultados

Para probar el correcto funcionamiento de la aplicación realizaremos distintas pruebas de validación. Realizaremos una batería de test de los aspectos más conflictivos en el uso de la aplicación, como puede ser el "login" del tribunal, entrada de estudiantes, etc. Cabe señalar que las pruebas realizadas se basan en simulaciones de lo que la aplicación podría realizar, son simulaciones porque no son datos reales, ya que no tenemos acceso a los mismos.

8.1 Login tribunal

En este apartado comprobaremos el correcto funcionamiento del "login" de la aplicación, el cual es necesario para comenzar su uso.

8.1.1 Prueba login 1

Comenzaremos mostrando el "login" correcto, para ello introducimos un usuario existente:

- Nombre de usuario: **Veronica**
- Contraseña: **tribunaluamveronica**

Figura 11: Login correcto 1

Tras introducir los datos indicados anteriormente y presionar el botón “Iniciar sesión”, pasamos a la siguiente pantalla que mostramos a continuación. Como podemos comprobar el “login” se ha realizado correctamente.

Figura 12: Login correcto 2

A continuación mostramos el caso de “login” incorrecto, insertando un usuario que no existe, por ejemplo:

- Nombre de usuario: **Leticia**
- Contraseña: **tribunaluamleticia**

Figura 13: Login incorrecto 1

Tras introducir los datos y presionar el botón para iniciar sesión, como en el caso anterior, nos muestra el siguiente aviso:

Figura 14: Login incorrecto 2

Como el usuario introducido no existe nos muestra un mensaje de error "*Invalid username or password*", para que corrijamos los datos insertados.

8.1.2 Prueba login 2

Por último mostraremos el caso en el que dejamos un campo incompleto:

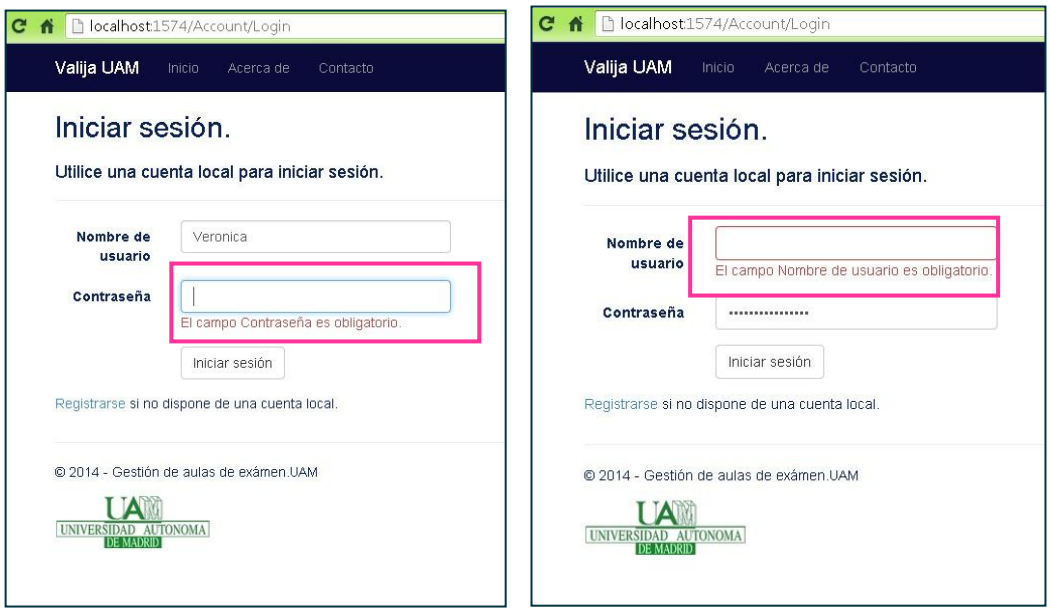


Figura 15: Login incompleto

Como podemos ver en las imágenes al dejar cualquiera de los campos incompletos se muestra un mensaje indicando que los campos son obligatorios.

Estas pruebas han tenido un resultado satisfactorio, ya que se ha podido comprobar que la aplicación resuelve favorablemente aquellos errores que se puedan comenter al realizar el "login" en la aplicación.

8.2 Entrada de estudiantes

8.2.1 Prueba Acceso 1

En el caso de que se tenga que introducir el carnet del estudiante a mano, por fallo hardware o por el mal estado del carnet del estudiante, se debe tener en cuenta la posibilidad de introducir el DNI incorrectamente.

Por ejemplo vamos a introducir un DNI inexistente en nuestra base de datos: "**000000000**"

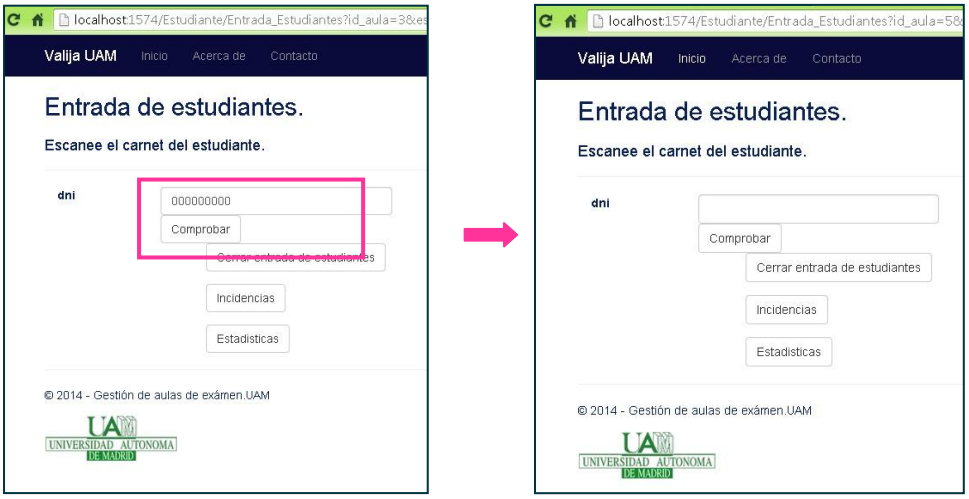


Figura 16: DNI incorrecto

Como podemos comprobar en la imagen superior, si el DNI introducido no se encuentra en la base de datos, la aplicación nos vuelve a solicitar el DNI del alumno.

8.2.2 Prueba Acceso 2

Otro caso a tener en cuenta, en la situación de que haya más de un aula abierta para la realización de una sesión, hay que evitar la posible duplicación de DNI, para que no sea posible que un estudiante esté en 2 aulas al mismo tiempo.

Para ello se abrirán 2 aulas para la sesión 1:

Insertamos al estudiante con DNI: "**333333333**", en el aula **1.**

Figura 17: Estudiante duplicado 1

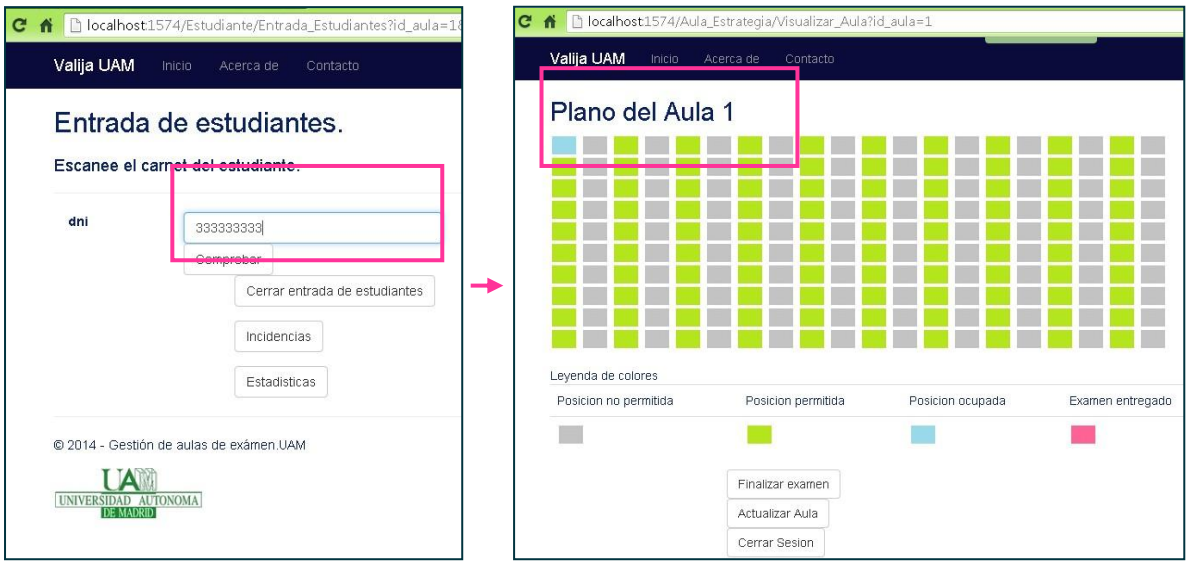


Figura 18: Estudiante duplicado 2

Ahora procedemos a abrir otro aula para la misma sesión, e intentaremos introducir al mismo estudiante con DNI "**333333333**"**,** en el aula **3.**

Figura 19: Estudiante duplicado 3

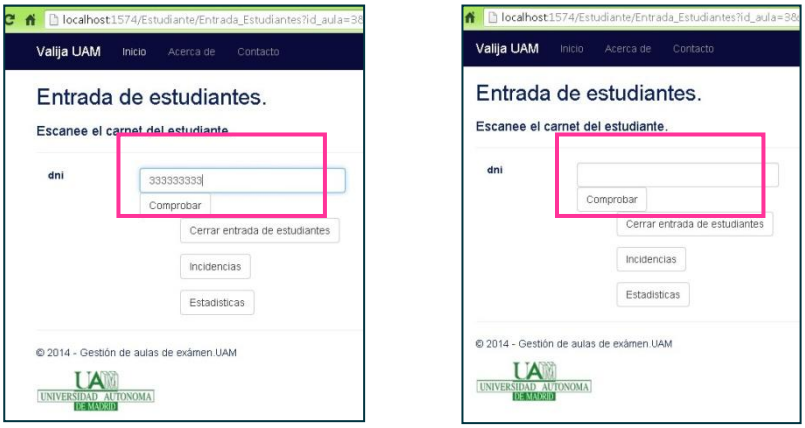


Figura 20: Estudiante duplicado 4

Figura 21: Estudiante duplicado 4

Como podemos observar en las imágenes posteriores, si intentamos introducir de nuevo el mismo estudiante, en una misma sesión en dos aulas distintas, nos vuelve a solicitar el DNI del estudiante, impidiendo que se puedan duplicar. Las figuras del plano del aula que mostramos confirman que en la segunda aula abierta,el aula 3, no se ha insertado el estudiante, puesto que el aula no tiene ninguna posición ocupada, mientras que en el aula 1, aparece la posición ocupada por el estudiante.

8.3 Cierre de sesión

8.3.1 Prueba Cierre 1

Para esta prueba vamos a comprobar que no se puede cerrar sesión hasta que todos los alumnos presentados entreguen sus exámenes.

En la siguiente imagen mostramos el aula con un número de estudiantes que están realizando el examen:

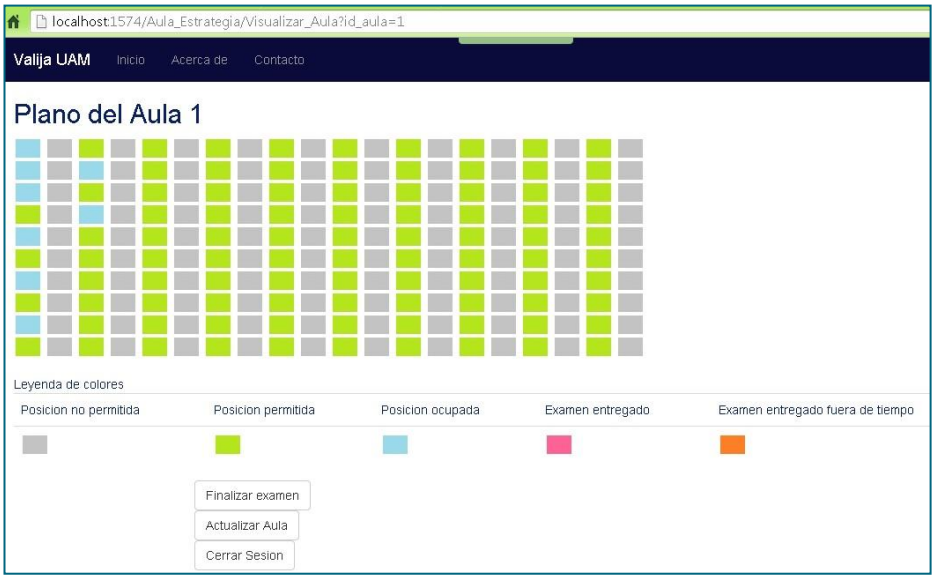


Imagen 22: Plano del aula con estudiantes

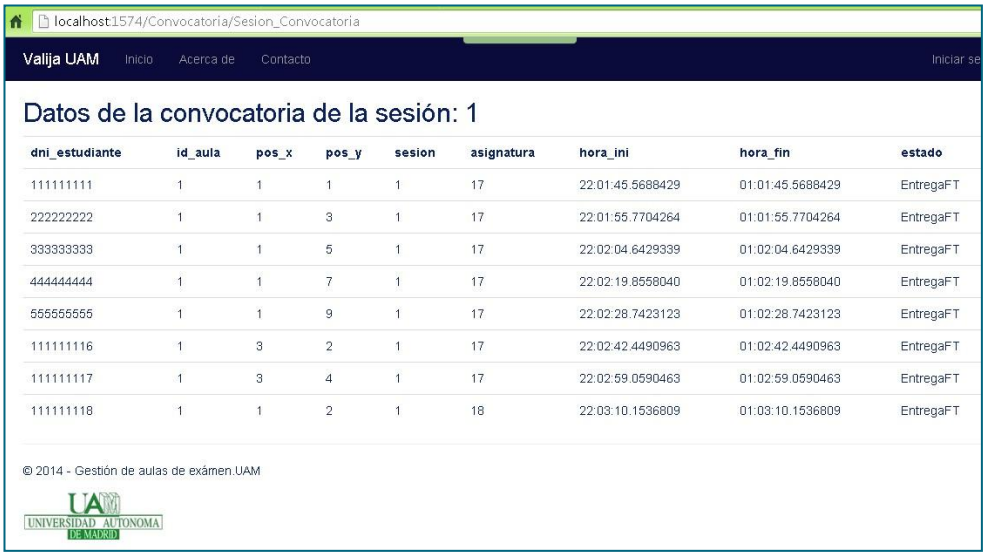
A continuación entregaremos todos los exámenes menos el de un estudiante:

Figura 23: Examen entregados 1

Ahora intentaremos finalizar sesión, mostrándonos la siguiente pantalla, puesto que hay un alumno que no ha entregado el examen:

Figura 24: Examen sin entregar

Para cerrar la sesión correctamente debemos realizar la entrega del examen pendiente, y finalmente aparecerá un resumen de la sesión:



dni_estudiante	id_aula	pos_x	pos_y	sesion	asignatura	hora_ini	hora_fin	estado
111111111	1	1	1	1	17	22:01:45.5688429	01:01:45.5688429	EntregaFT
222222222	1	1	3	1	17	22:01:55.7704264	01:01:55.7704264	EntregaFT
333333333	1	1	5	1	17	22:02:04.6429339	01:02:04.6429339	EntregaFT
444444444	1	1	7	1	17	22:02:19.8558040	01:02:19.8558040	EntregaFT
555555555	1	1	9	1	17	22:02:28.7423123	01:02:28.7423123	EntregaFT
111111116	1	3	2	1	17	22:02:42.4490963	01:02:42.4490963	EntregaFT
111111117	1	3	4	1	17	22:02:59.0590463	01:02:59.0590463	EntregaFT
111111118	1	1	2	1	18	22:03:10.1536809	01:03:10.1536809	EntregaFT

Figura 25: Cierre sesión

9 Conclusiones

El objetivo principal de este proyecto era implementar una aplicación para facilitar la gestión de las aulas en el desarrollo de exámenes, teniendo en cuenta restricciones como: permitir en una sesión múltiples asignaturas de distintos cursos, admitir estudiantes libremente en diferentes aulas y garantizando la seguridad frente a copias al ubicar estudiantes de materias afines suficientemente separados de acuerdo con las restricciones de aula impuestas por el tribunal. Para ello basamos el estudio para obtener los requisitos funcionales, en los pasos que serían necesarios para:

- Iniciar una sesión de examen
  - Login del tribunal
  - Apertura de la sesión correspondiente
  - Selección del aula
  - Elección de la estrategia para la disposición en el aula),
- Entrada de estudiantes
  - Identificación del alumno
  - Selección de la asignatura a examinar
  - Impresión de examen
  - Ubicación del alumno en el aula
  - Almacenar datos del alumno
- Salida de estudiantes
  - Identificación del alumno
  - Actualización del aula de examen
  - Actualización de los datos de convocatoria

A partir de estos pasos se ha podido deducir una mayor funcionalidad, como la inserción de incidencias, la visualización del aula, etc.

Los problemas surgidos a lo largo del proyecto han estado relacionados en su mayor parte con el aprendizaje del entorno .NET y el uso de un framework cuya última versión es aún muy reciente ya que a lo largo de la carrera no se ha visto dicho entorno, por lo que

han podido hacer más costoso el desarrollo de ésta aplicación, pero al mismo tiempo estos problemas han hecho que se adquiera un mayor conocimiento sobre este tema.

Tras todo el esfuerzo y trabajo realizado para el desarrollo de este proyecto, consideremos que se ha obtenido un buen resultado, cumpliendo las expectativas esperadas.

Hay que destacar que con el desarrollo de éste proyecto he podido poner en práctica todo lo aprendido a lo largo de estos años en la universidad, desde el modelado del software desarrollado, ya que me ha permitido visualizar el sistema a construir, además de saber extraer los requisitos la aplicación, pudiendo así mimetizar el funcionamiento que debía tener la aplicación, pasando también por las fases de planificación, diseño e implementación de la misma, estudiadas en ingeniería del software, añadiendo también el desarrollo de documentos, como el de la presente memoria.

Por otra parte se han podido aplicar los conocimientos adquiridos sobre bases de datos, permitiendo escoger los tipos adecuados para cada atributo, así como crear las consultas necesarias para el funcionamiento de la aplicación

Desde un punto de vista personal, creo que con este proyecto he podido demostrar todo lo aprendido en la carrera, aparte de poner a prueba los conocimientos adquiridos como ingeniera, creyendo personalmente que el trabajo obtenido ha sido muy satisfactorio.

10 Líneas futuras

En este capítulo vamos a abordar aquellos aspectos que se podrán realizar en un futuro para mejorar y aumentar la funcionalidad de la aplicación, ya que está pensado que ésta pueda ser utilizada en un ámbito real con posterioridad.

A continuación definimos las posibles mejoras:

- Imprimir el examen con los datos del alumno en cuestión. Una de las mejoras planificadas tiene que ver con la impresión de la hoja de examen personalizada. Cuando el estudiante accede al aula y se lee su carnet se comprueba qué asignatura debe examinarse y se imprime en ese instante el examen (previamente encriptado por el equipo docente) con la cabecera con los datos personales del estudiante.
- Imprimir automáticamente el examen tras pulsar el botón “Imprimir”. En la actual versión de la aplicación, al presionar el botón se abre el examen con un lector de pdf, a través del cual se imprime el examen. Con esta mejora se pretende ahorrar este paso para mejorar la eficiencia de la aplicación.
- Implantar un reloj para que la entrada de estudiantes se pueda realizar durante 30 minutos, o más tiempo dependiendo si la entrada de estudiantes al aula es lenta o fluida, en el caso de que sea lenta este tiempo podrá ser mayor. Pasado este tiempo la entrada de estudiantes al aula se cerrará automáticamente.
- Seleccionar posiciones prohibidas en el aula manualmente. Con esta mejora se pretende permitir a los miembros del tribunal, establecer posiciones no permitidas dentro del aula manualmente, además de las establecidas por la estrategia escogida.
- Seleccionar posición del alumno en el aula manualmente. Al igual que en el punto anterior, esta mejora permite establecer al tribunal una posición ya este permitida o no por la estrategia, para ubicar al alumno en el aula.
- Realizar la gestión sobre las asignaturas de reserva. Con esta mejora, se pretende realizar el funcionamiento relacionado con la gestión de las

asignaturas en reserva, de los alumnos a los que les han coincidido más de una asignatura.

- Mostrar información al pasar por los asientos del aula. Esta mejora está relacionada con la visualización del aula de examen, su función será mostrar los datos del estudiante ubicado en la posición seleccionada.
- Imprimir justificante para el alumno. Esta mejora aporta una funcionalidad extra, para poder imprimir un justificante al alumno que lo solicite al entregar al examen.
- Mejorar la interfaz de usuario. Con esta mejora se pretende obtener una interfaz de usuario más atractiva.

11 Referencias

[1] http://www.dicub.es/i15/Noticias.aspx?an=0&mes=0&apli=18

[2] http://www.dicub.es/p18/Valija-Virtual.aspx

[3] http://www.php.net/manual/es/features.php

[4] http://msdn.microsoft.com/es-es/library/4w3ex9c2(v=vs.100).aspx

[5] http://es.wikipedia.org/wiki/ASP.NET_MVC_Framework

[6] http://www.visualstudio.com/

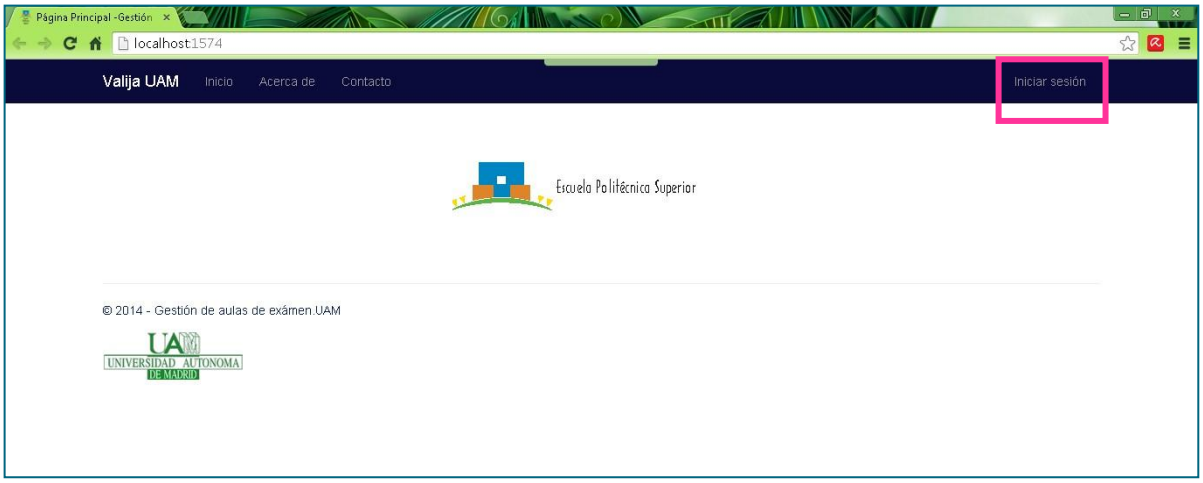
12 Anexos

Manual de usuario/Interfaz

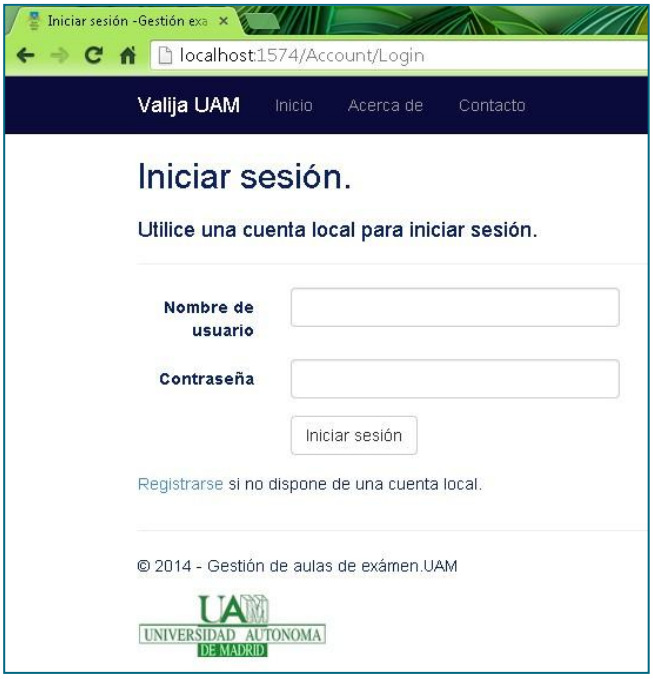
A continuación vamos a describir el manual de usuario de la aplicación, es probable que la interfaz de la misma varíe con respecto a la que se presentará en la exposición.

A continuación vamos a describir el manual de usuario de la aplicación, es probable que la interfaz de la misma varíe con respecto a la que se presentará en la exposición.

- **Iniciar sesión en la aplicación**



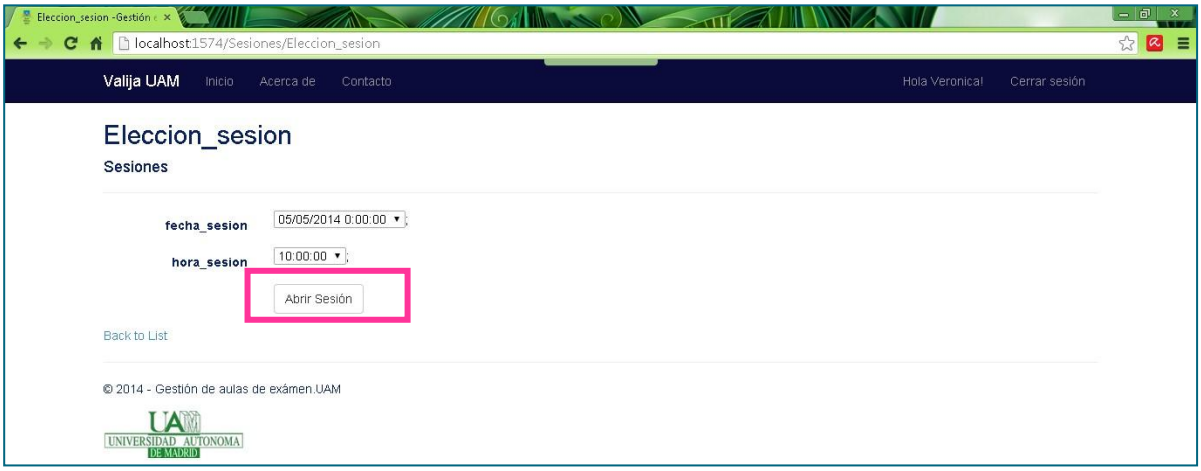
Se debe presionar sobre el botón “Iniciar sesión”. A continuación aparecerá el siguiente formulario:



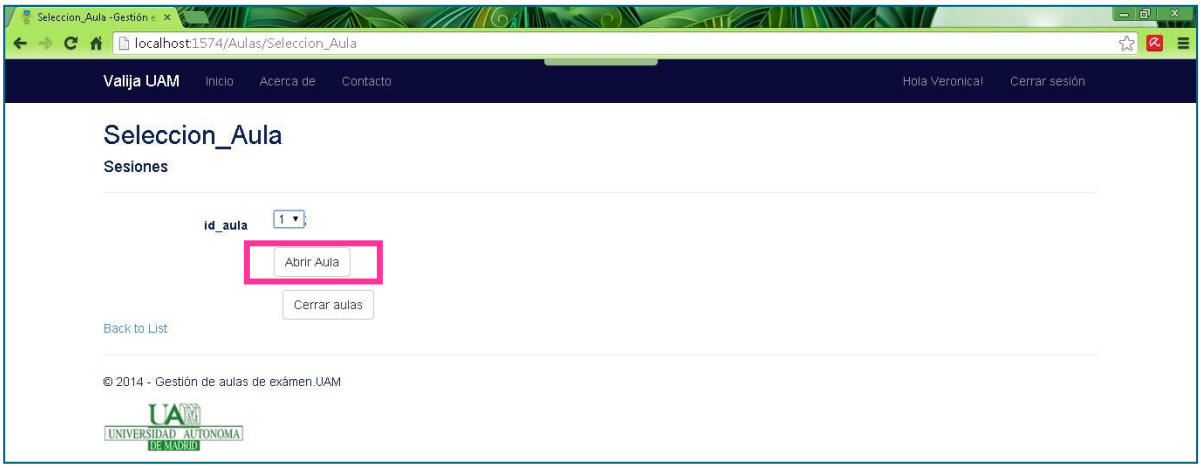
En este momento se debe introducir el nombre de usuario y la contraseña del mismo y presionar el botón “Iniciar sesión”:

- **Iniciar sesión de examen:**

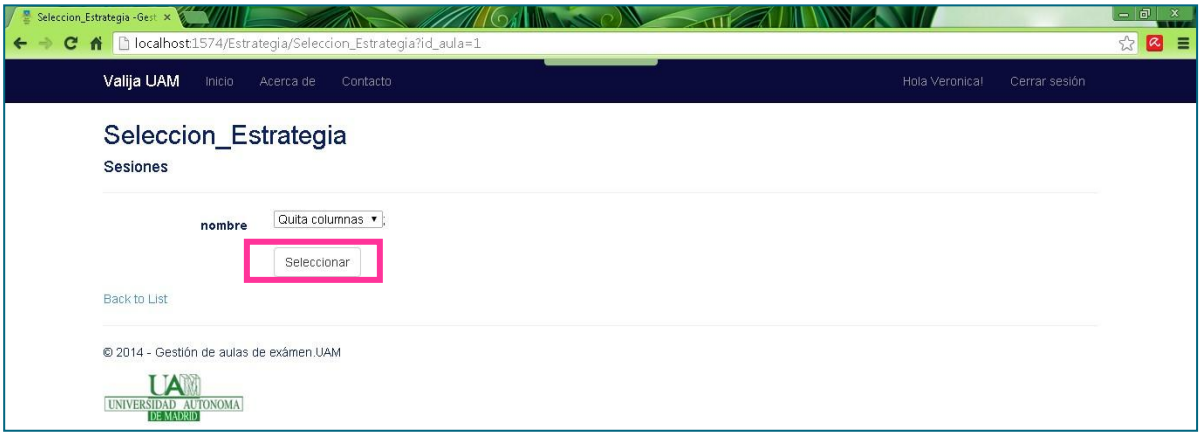
Una vez iniciada la sesión del tribunal, se procede a abrir la sesión de examen a realizar, cuando se haya seleccionado el día y hora de la sesión se debe pulsar el botón “Abrir Sesión”



A continuación, pasamos a seleccionar el aula donde se realizarán los exámenes, una vez seleccionado pulsaremos sobre “Abrir aula”



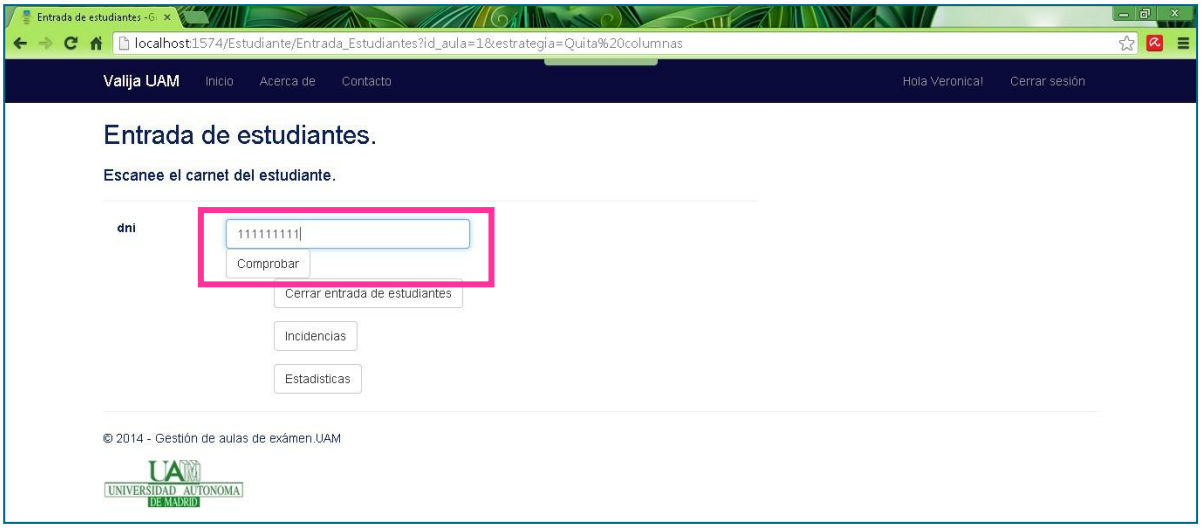
Ahora se debe elegir la estrategia para distribuir a los alumnos en el aula, al igual que para el aula, una vez tengamos seleccionada la estrategia pulsaremos “Seleccionar”.



Hasta aquí ya están hechos todos los pasos para abrir la sesión de exámenes, a partir de este momento se produce la entrada de estudiantes.

- **Entrada de estudiantes a la sesión**

En la siguiente pantalla se debe escanear el carnet del estudiante, para insertar su DNI en el formulario, quedaría de la siguiente forma:

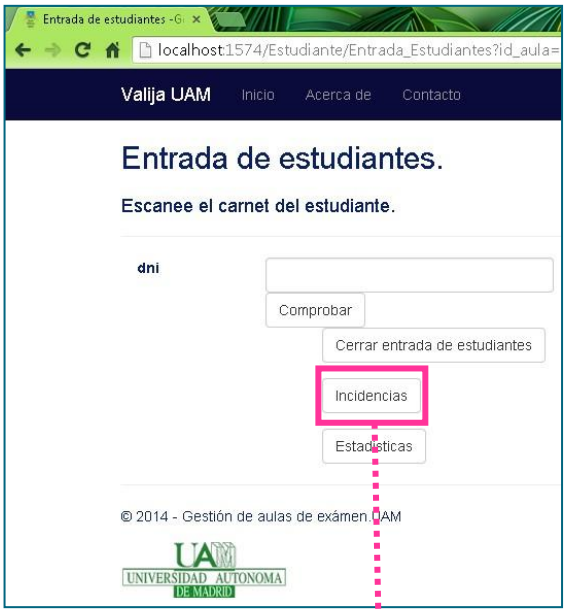


Al pulsar el botón “Comprobar”, aparecerán las asignaturas de las que se puede examinar el alumno en esta sesión:

Para imprimir el examen del alumno presionaremos el botón “Imprimir”, tras el cual se abrirá el “pdf” del examen el cual deberemos imprimir.

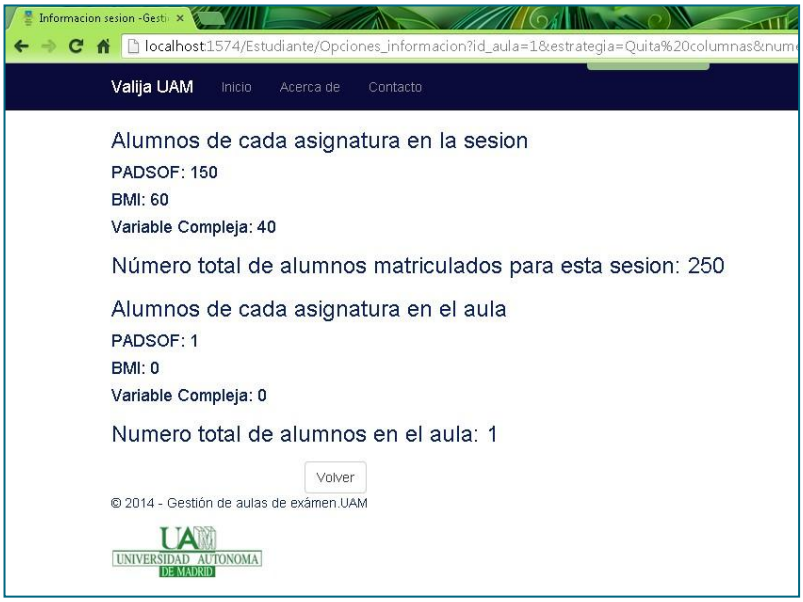
Tras esta acción, la aplicación nos vuelve a dirigir a la pantalla de entrada de estudiantes. Desde esta pantalla podremos escribir una nueva incidencia o conocer el número de asistentes a la sesión.

En el caso de escoger incidencias:


Entrada de estudiantes -G
localhost:1574/Estudiante/Entrada_Estudiantes?id_aula=
Valija UAM
Inicio
Acerca de
Contacto
Entrada de estudiantes.
Escanee el carnet del estudiante.
dni
Comprobar
Cerrar entrada de estudiantes
Incidencias
Estadísticas
© 2014 - Gestión de aulas de exámen.UAM
UNIVERSIDAD AUTONOMA DE MADRID


Index -Gestión examenes
localhost:1574/Incidencias?id_aula=1&estrategia=Quita%20columnas&numero=1
Valija UAM
Inicio
Acerca de
Contacto
Hola Veronica!
Cerrar sesión
Index
Create New
nombre
tipo
implicados
descripcion
id_sesion
Volver
© 2014 - Gestión de aulas de exámen.UAM
UNIVERSIDAD AUTONOMA DE MADRID

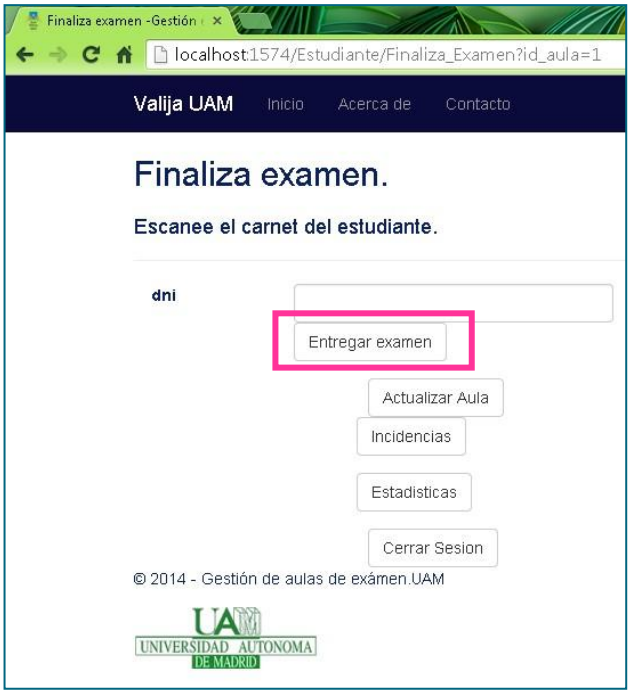
Si elegimos estadísticas:



Tras la entrada de los estudiantes a la sesión pasamos a abrir el proceso para la entrega de exámenes.

- **Cierre entrada de estudiantes**

Para pasar a esta pantalla debemos presionar el botón “Cerrar entrada estudiantes”, el cual nos llevará a la siguiente pantalla:

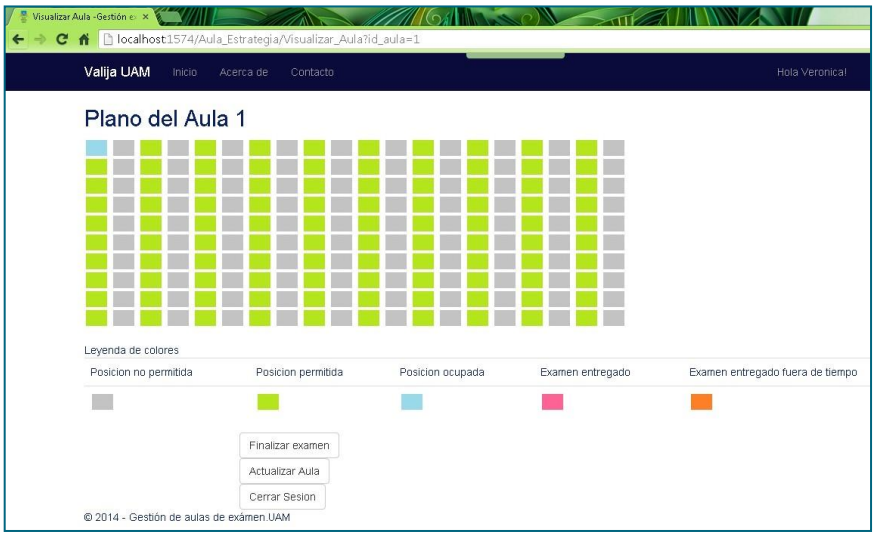


Para realizar la entrega de un examen, se deberá escanear el código pegado en el examen del estudiante, una vez leído presionaremos el botón “Entregar examen”, el cual nos mostrará la siguiente información:

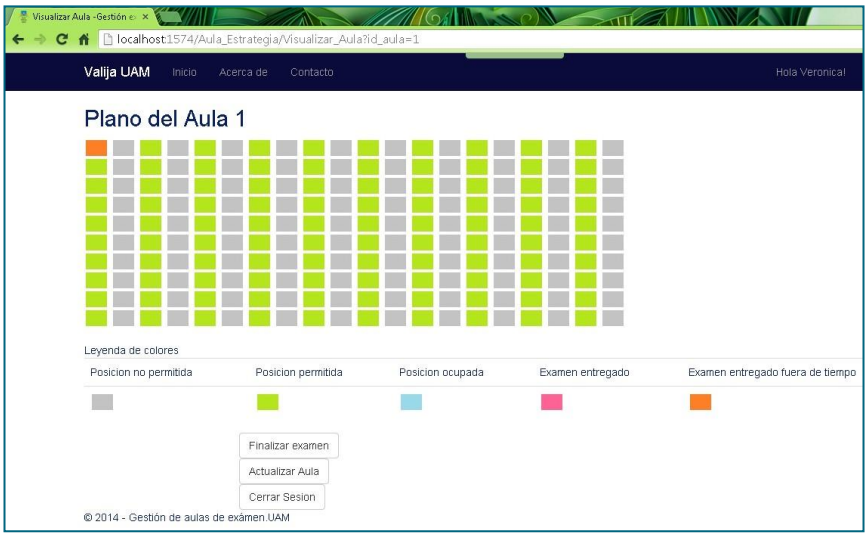
Cuando se pulsa el botón “Entregado”, nos reconduce a la página anterior.

Si se pulsa “Actualizar aula”, se muestra un plano del aula, con las posiciones libres, ocupadas, permitidas y no permitidas.

Haciendo el examen:



Con examen entregado:



Por último nos queda cerrar la sesión de exámenes.

- **Finalizar sesión de examen**

Para ello presionaremos el botón “Cerrar sesión”, el cual nos mostrará un resumen de los datos de la convocatoria: